**Section 1**

**Sum of three numbers**

a = int(input()) b = int(input()) c = int(input()) print(a + b + c)

**Area of right-angled triangle**

a = int(input()) b = int(input()) print(a \* b / 2)

**Apple sharing**

n = int(input()) k = int(input()) print(k // n) print(k % n)

**Digital clock**

n = int(input()) hours = n // 60 minutes = n % 60 print(hours, minutes)

**Hello, Harry!**

print('Hello, ' + input() + '!')

**Previous and next**

n = int(input())

print('The next number for the number ' + str(n) + ' is ' + str(n + 1) + '.')

print('The previous number for the number ' + str(n) + ' is ' + str(n - 1) + '.')

**School desks**

a = int(input()) b = int(input()) c = int(input())

print(a // 2 + b // 2 + c // 2 + a % 2 + b % 2 + c % 2)

**Section 2**

**Minimum of two numbers**

a=int(input()) b=int(input()) if a<b:

print(a) else:

print(b)

**Sign function**

x = int(input())

if x > 0:

print(1) elif x == 0:

print(0) else:

print(-1)

**Minimum of three numbers**

a = int(input()) b = int(input()) c = int(input()) if b >= a <= c: print(a)

elif a >= b <= c: print(b)

else:

print(c)

**Leap year**

year = int(input())

if (year % 4 == 0) and (year % 100 != 0) or (year % 400 == 0): print('LEAP')

else:

print('COMMON')

**Equal numbers**

a = int(input()) b = int(input()) c = int(input()) if a == b == c: print(3)

elif a == b or b == c or a == c: print(2)

else:

print(0)

**Rook move**

x1 = int(input()) y1 = int(input()) x2 = int(input())

y2 = int(input())if x1 == x2 or y1 == y2: print('YES')

else:

print('NO')

**Chess board**

x1 = int(input()) y1 = int(input()) x2 = int(input()) y2 = int(input())

if (x1 + y1 + x2 + y2) % 2 == 0:

print('YES') else:

print('NO')

**King move**

x1 = int(input()) y1 = int(input()) x2 = int(input()) y2 = int(input())

if abs(x1 - x2) <= 1 and abs(y1 - y2) <= 1: print('YES')

else:

print('NO')

**Bishop moves**

x1 = int(input()) y1 = int(input()) x2 = int(input()) y2 = int(input())

if abs(x1 - x2) == abs(y1 - y2): print('YES')

else:

print('NO')

**Queen move**

x1 = int(input()) y1 = int(input()) x2 = int(input()) y2 = int(input())

if abs(x1 - x2) == abs(y1 - y2) or x1 == x2 or y1 == y2: print('YES')

else:

print('NO')

**Knight move**

x1 = int(input()) y1 = int(input()) x2 = int(input()) y2 = int(input()) dx = abs(x1 - x2) dy = abs(y1 - y2)

if dx == 1 and dy == 2 or dx == 2 and dy == 1: print('YES')

else:

print('NO')

**Chocolate bar**

n = int(input()) m = int(input()) k = int(input())

if k < n \* m and ((k % n == 0) or (k % m == 0)): print('YES')

else:

print('NO')

**Section 3**

**Last digit of integer**

a = int(input())

print(a % 10)

**Fractional part**

x = float(input())

print(x - int(x))

**First digit after decimal point**

x = float(input())

print(int(x \* 10) % 10)

**Car route**

from math import ceil

n = int(input()) m = int(input()) print(ceil(m / n))

**Total cost**

a = int(input()) b = int(input()) n = int(input())

cost = n \* (100 \* a + b) print(cost // 100, cost % 100)

**Tens digit**

n = int(input())

print(n // 10 % 10)

**Sum of digits**

n = int(input()) a = n // 100

b = n // 10 % 10 c = n % 10 print(a + b + c)

**Clock face – 1**

h = int(input()) m = int(input()) s = int(input())

print(h \* 30 + m \* 30 / 60 + s \* 30 / 3600)

**Clock face – 2**

alpha = float(input()) print(alpha % 30 \* 12)

**Section 4**

**Series – 1**

a = int(input()) b = int(input())

for i in range(a, b + 1): print(i)

**Series – 2**

a = int(input()) b = int(input()) if a < b:

for i in range(a, b + 1): print(i)

else:

for i in range(a, b - 1, -1): print(i)

**Sum of ten numbers**

res = 0

for i in range(10): res += int(input())

print(res)

**Sum of N numbers**

n = int(input()) res = 0

for i in range(n):

res += int(input()) print(res)

**Sum of cubes**

res = 0

for i in range(1, int(input()) + 1): res += i \*\* 3

print(res)

**Factorial**

res = 1

n = int(input())

for i in range(1, n + 1): res \*= i

print(res)

**The number of zeros**

num\_zeroes = 0

for i in range(int(input())): if int(input()) == 0:

num\_zeroes += 1 print(num\_zeroes)

**Adding factorials**

n = int(input()) partial\_factorial = 1

partial\_sum = 0

for i in range(1, n + 1): partial\_factorial \*= i partial\_sum += partial\_factorial

print(partial\_sum)

**Lost card**

n = int(input()) sum\_cards = 0

for i in range(1, n + 1): sum\_cards += i

# One can prove the following:

# sum\_cards == n \* (n + 1) // 2

# However, we'll calculate that using the loop. for i in range(n - 1):

sum\_cards -= int(input()) print(sum\_cards)

**Ladder**

n = int(input())

for i in range(1, n + 1): for j in range(1, i + 1):

print(j, sep='', end='') print()

**Section 5 Slices**

s = input() print(s[2])

print(s[-2])

print(s[:5])

print(s[:-2])

print(s[::2])

print(s[1::2])

print(s[::-1])

print(s[::-2]) print(len(s))

**The number of words**

print(input().count(' ') + 1)

**The two halves**

s = input()

print(s[(len(s) + 1) // 2:] + s[:(len(s) + 1) // 2])

**To swap the two words**

s = input()

first\_word = s[:s.find(' ')] second\_word = s[s.find(' ') + 1:] print(second\_word + ' ' + first\_word)

**The first and last occurrence**

s = input()

if s.count('f') == 1:

print(s.find('f')) elif s.count('f') >= 2:

print(s.find('f'), s.rfind('f'))

**The second occurrence**

s = input()

if s.count('f') == 1: print(-1)

elif s.count('f') < 1: print(-2)

else:

print(s.find('f', s.find('f') + 1))

**Remove the fragment**

s = input()

s = s[:s.find('h')] + s[s.rfind('h') + 1:] print(s)

**Reverse the fragment**

s = input()

a = s[:s.find('h')]

b = s[s.find('h'):s.rfind('h') + 1] c = s[s.rfind('h') + 1:]

s = a + b[::-1] + c print(s)

**Replace the substring**

print(input().replace('1', 'one'))

**Delete a character**

print(input().replace('@', ''))

**Replace within the fragment**

s = input()

a = s[:s.find('h') + 1]

b = s[s.find('h') + 1:s.rfind('h')] c = s[s.rfind('h'):]

s = a + b.replace('h', 'H') + c print(s)

**Delete every third character**

s = input() t = ''

for i in range(len(s)): if i % 3 != 0:

t = t + s[i] print(t)

**Section 6**

**List of squares**

n = int(input()) i = 1

while i \*\* 2 <= n: print(i \*\* 2)

i += 1

**Least divisor**

n = int(input()) i = 2

while n % i != 0: i += 1

print(i)

**The power of two**

n = int(input()) two\_in\_power = 2

power = 1

while two\_in\_power <= n: two\_in\_power \*= 2

power += 1

print(power - 1, two\_in\_power // 2)

**Morning jog**

x = int(input()) y = int(input()) i = 1

while x < y:

x \*= 1.1

i += 1

print(i)

**The length of the sequence**

len = 0

while int(input()) != 0: len += 1

print(len)

**The sum of the sequence**

sum = 0

element = int(input()) while element != 0:

sum += element

element = int(input()) print(sum)

**The average of the sequence**

sum = 0

len = 0

element = int(input()) while element != 0:

sum += element len += 1

element = int(input()) print(sum / len)

**The maximum of the sequence**

max = 0

element = -1

while element != 0: element = int(input()) if element > max:

max = element print(max)

**The index of the maximum of a sequence**

max = 0

index\_of\_max = -1

element = -1

len = 1

while element != 0: element = int(input()) if element > max:

max = element index\_of\_max = len

len += 1 print(index\_of\_max)

**The number of even elements of the sequence**

num\_even = -1

element = -1

while element != 0: element = int(input()) if element % 2 == 0:

num\_even += 1 print(num\_even)

**The number of elements that are greater than the previous one**

prev = int(input()) answer = 0

while prev != 0:

next = int(input())

if next != 0 and prev < next: answer += 1

prev = next print(answer)

**The second maximum**

first\_max = int(input()) second\_max = int(input()) if first\_max < second\_max:

first\_max, second\_max = second\_max, first\_max element = int(input())

while element != 0:

if element > first\_max:

second\_max, first\_max = first\_max, element elif element > second\_max:

second\_max = element element = int(input())

print(second\_max)

**The number of elements equal to the maximum**

maximum = 0

num\_maximal = 0

element = -1

while element != 0: element = int(input()) if element > maximum:

maximum, num\_maximal = element, 1 elif element == maximum:

num\_maximal += 1 print(num\_maximal)

**Fibonacci numbers**

n = int(input()) if n == 0:

print(0) else:

a, b = 0, 1

for i in range(2, n + 1): a, b = b, a + b

print(b)

**The index of a Fibonacci number**

a = int(input()) if a == 0:

print(0) else:

fib\_prev, fib\_next = 0, 1

n = 1

while fib\_next <= a: if fib\_next == a:

print(n) break

fib\_prev, fib\_next = fib\_next, fib\_prev + fib\_next n += 1

else:

print(-1)

**The maximum number of**

**consecutive equal elements**

prev = -1

curr\_rep\_len = 0

max\_rep\_len = 0 element = int(input()) while element != 0:

if prev == element: curr\_rep\_len += 1

else:

prev = element

max\_rep\_len = max(max\_rep\_len, curr\_rep\_len) curr\_rep\_len = 1

element = int(input())

max\_rep\_len = max(max\_rep\_len, curr\_rep\_len)

print(max\_rep\_len)

**Section 7 Even indices**

a = input().split()

for i in range(0, len(a), 2): print(a[i])

**Even elements**

a = [int(i) for i in input().split()] for elem in a:

if elem % 2 == 0: print(elem)

**Greater than the previous**

a = [int(i) for i in input().split()] for i in range(1, len(a)):

if a[i] > a[i - 1]: print(a[i])

**Neighbours of the same sign**

a = [int(i) for i in input().split()] for i in range(1, len(a)):

if a[i - 1] \* a[i] > 0:

print(a[i - 1], a[i]) break

**Greater than neighbours**

a = [int(i) for i in input().split()] counter = 0

for i in range(1, len(a) - 1):

if a[i - 1] < a[i] > a[i + 1]: counter += 1

print(counter)

**The largest element**

index\_of\_max = 0

a = [int(i) for i in input().split()] for i in range(1, len(a)):

if a[i] > a[index\_of\_max]: index\_of\_max = i

print(a[index\_of\_max], index\_of\_max)

**The number of distinct elements**

a = [int(i) for i in input().split()] num\_distinct = 1

for i in range(0, len(a) - 1): if a[i] != a[i + 1]:

num\_distinct += 1 print(num\_distinct)

**Swap neighbours**

a = [int(i) for i in input().split()] for i in range(1, len(a), 2):

a[i - 1], a[i] = a[i], a[i - 1]

print(' '.join([str(i) for i in a]))

**Swap min and max**

a = [int(s) for s in input().split()] index\_of\_min = 0

index\_of\_max = 0

for i in range(1, len(a)):

if a[i] > a[index\_of\_max]: index\_of\_max = i

if a[i] < a[index\_of\_min]: index\_of\_min = i

a[index\_of\_min], a[index\_of\_max] = a[index\_of\_max], a[index\_of\_min]

print(' '.join([str(i) for i in a]))

**The number of pairs of equal**

a = [int(s) for s in input().split()] counter = 0

for i in range(len(a)):

for j in range(i + 1, len(a)): if a[i] == a[j]:

counter += 1 print(counter)

**Unique elements**

a = [int(s) for s in input().split()] for i in range(len(a)):

for j in range(len(a)):

if i != j and a[i] == a[j]: break

else:

print(a[i], end=' ')

**Queens**

n = 8 x = []

y = []

for i in range(n):

new\_x, new\_y = [int(s) for s in input().split()] x.append(new\_x)

y.append(new\_y) correct = True

for i in range(n):

for j in range(i + 1, n):

if x[i] == x[j] or y[i] == y[j] or abs(x[i] - x[j]) == abs(y[i] - y[j]):

correct = False

if correct:

print('NO') else:

print('YES')

**The bowling alley**

n, k = [int(s) for s in input().split()] bahn = ['I'] \* n

for i in range(k):

left, right = [int(s) for s in input().split()] for j in range(left - 1, right):

bahn[j] = '.'

print(''.join(bahn))

**Section 8**

**The length of the segment**

from math import sqrt

def distance(x1, y1, x2, y2):

return sqrt((x1 - x2) \*\* 2 + (y1 - y2) \*\* 2)

x1 = float(input()) x2 = float(input()) y1 float(input()) y2 = float(input())

print(distance(x1, x2, y1, y2))

**Negative exponent**

def power(a, n): res = 1

for i in range(abs(n)): res \*= a

if n >= 0:

return res else:

return 1 / res

print(power(float(input()), int(input())))

**Uppercase**

def capitalize(word): first\_letter\_small = word[0]

first\_letter\_big = chr(ord(first\_letter\_small) - ord('a') + ord('A'))

return first\_letter\_big + word[1:]

source = input().split() res = []

for word in source: res.append(capitalize(word))

print(' '.join(res))

**Exponentiation**

def power(a, n):

if n == 0:

return 1 else:

return a \* power(a, n - 1)

print(power(float(input()), int(input())))

**Reverse the sequence**

def reverse():

x = int(input()) if x != 0:

reverse() print(x)

reverse()

**Fibonacci numbers**

def fib(n):

if n == 1 or n == 2: return 1

else:

return fib(n - 1) + fib(n - 2)

print(fib(int(input())))

**Section 9 Maximum**

n, m = [int(i) for i in input().split()]

a = [[int(j) for j in input().split()] for i in range(n)] best\_i, best\_j = 0, 0

curr\_max = a[0][0] for i in range(n):

for j in range(m):

if a[i][j] > curr\_max: curr\_max = a[i][j] best\_i, best\_j = i, j

print(best\_i, best\_j)

**Snowflake**

n = int(input())

a = [['.'] \* n for i in range(n)] for i in range(n):

a[i][i] = '\*'

a[n // 2][i] = '\*'

a[i][n // 2] = '\*'

a[i][n - i - 1] = '\*' for row in a:

print(' '.join(row))

n, m = [int(i) for i in input().split()] a = []

for i in range(n): a.append([])

for j in range(m):

if (i + j) % 2 == 0: a[i].append('.')

else:

a[i].append('\*') for row in a:

print(' '.join(row))

**Chess board**

**The diagonal parallel to the main**

n = int(input())

a = [[abs(i - j) for j in range(n)] for i in range(n)] for row in a:

print(' '.join([str(i) for i in row]))

**side diagonal**

n = int(input())

a = [[0] \* n for i in range(n)] for i in range(n):

a[i][n - i - 1] = 1 for i in range(n):

for j in range(n - i, n): a[i][j] = 2

for row in a:

for elem in row: print(elem, end=' ')

print()

**Swap the columns**

def swap\_columns(a, i, j): for k in range(len(a)):

a[k][i], a[k][j] = a[k][j], a[k][i]

n, m = [int(i) for i in input().split()]

a = [[int(j) for j in input().split()] for i in range(n)] i, j = [int(i) for i in input().split()]

swap\_columns(a, i, j)

print('\n'.join([' '.join([str(i) for i in row]) for row in a]))

**Scale a matrix**

m, n = [int(k) for k in input().split()]

A = [[int(k) for k in input().split()] for i in range(m)] c = int(input())

for i in range(m):

for j in range(n):

A[i][j] \*= c

print('\n'.join([' '.join([str(k) for k in row]) for row in A]))

**Multiply two matrices**

m, n, r = [int(k) for k in input().split()]

A = [[int(k) for k in input().split()] for i in range(m)] B = [[int(k) for k in input().split()] for j in range(n)] C = [[0]\*r for i in range(m)]

for i in range(m): for k in range(r):

for j in range(n):

C[i][k] += A[i][j] \* B[j][k]

print('\n'.join([' '.join([str(k) for k in row]) for row in C]))

**Section 10**

**The number of distinct numbers**

print(len(set(input().split())))

**The number of equal numbers**

print(len(set(input().split()) & set(input().split())))

**The intersection of sets**

print(\*sorted(set(input().split()) & set(input().split()), key=int))

**Has the number been encountered before**

numbers = [int(s) for s in input().split()] occur\_before = set()

for num in numbers:

if num in occur\_before: print('YES')

else:

print('NO') occur\_before.add(num)

**Cubes**

def print\_set(some\_set): print(len(some\_set))

print(\*[str(item) for item in sorted(some\_set)])

N, M = [int(s) for s in input().split()] A\_colors, B\_colors = set(), set()

for i in range(N): A\_colors.add(int(input()))

for i in range(M): B\_colors.add(int(input()))

print\_set(A\_colors & B\_colors) print\_set(A\_colors - B\_colors) print\_set(B\_colors - A\_colors)

**The number of distinct words in some text**

words = set()

for \_ in range(int(input())): words.update(input().split())

print(len(words))

**Guess the number**

n = int(input())

all\_nums = set(range(1, n + 1)) possible\_nums = all\_nums

while True:

guess = input()

if guess == 'HELP': break

guess = {int(x) for x in guess.split()} answer = input()

if answer == 'YES': possible\_nums &= guess

else:

possible\_nums &= all\_nums - guess

print(' '.join([str(x) for x in sorted(possible\_nums)]))

**Polyglots**

students = [{input() for j in range(int(input()))} for i in range(int(input()))]

known\_by\_everyone, known\_by\_someone = set.intersection(\*students), set.union(\*students) print(len(known\_by\_everyone), \*sorted(known\_by\_everyone), sep='\n')

print(len(known\_by\_someone), \*sorted(known\_by\_someone), sep='\n')

**Section 11**

**Number of occurrences**

counter = {}

for word in input().split():

counter[word] = counter.get(word, 0) + 1 print(counter[word] - 1, end=' ')

**Dictionary of synonyms**

n = int(input()) d = {}

for i in range(n):

first, second = input().split() d[first] = second

d[second] = first print(d[input()])

**Elections in the USA**

num\_votes = {}

for \_ in range(int(input())): candidate, votes = input().split()

num\_votes[candidate] = num\_votes.get(candidate, 0) + int(votes)

for candidate, votes in sorted(num\_votes.items()): print(candidate, votes)

**The most frequent word**

counter = {}

for i in range(int(input())): line = input().split() for word in line:

counter[word] = counter.get(word, 0) + 1

max\_count = max(counter.values())

most\_frequent = [k for k, v in counter.items() if v == max\_count]

print(min(most\_frequent))

**Access rights**

OPERATION\_PERMISSION = {

'read': 'R',

'write': 'W',

'execute': 'X',

}

file\_permissions = {}

for i in range(int(input())):

file, \*permissions = input().split() file\_permissions[file] = set(permissions)

for i in range(int(input())): operation, file = input().split()

if OPERATION\_PERMISSION[operation] in file\_permissions[file]:

print('OK') else:

print('Access denied')

**Countries and cities**

motherland = {}

for i in range(int(input())):

country, \*cities = input().split() for city in cities:

motherland[city] = country

for i in range(int(input())): print(motherland[input()])

**Frequency analysis**

from collections import Counter

words = []

for \_ in range(int(input())): words.extend(input().split())

counter = Counter(words)

pairs = [(-pair[1], pair[0]) for pair in counter.most\_common()] words = [pair[1] for pair in sorted(pairs)] print('\n'.join(words))

# You can also solve this problem without Counter: #

# n = int(input()) # counts = {}

# for \_ in range(n):

# for word in input().split():

# counts[word] = counts.get(word, 0) + 1 #

# freqs = [(-count, word) for (word, count) in counts.items()] # for c, word in sorted(freqs):

# print(word)

**English-Latin dictionary**

from collections import defaultdict

latin\_to\_english = defaultdict(list) for i in range(int(input())):

english\_word, latin\_translations\_chunk = input().split(' -

')

latin\_translations = latin\_translations\_chunk.split(', ') for latin\_word in latin\_translations:

latin\_to\_english[latin\_word].append(english\_word)

print(len(latin\_to\_english))

for latin\_word, english\_translations in sorted(latin\_to\_english.items()):

print(latin\_word + ' - ' + ', '.join(english\_translations))