import math, re, numpy, pandas  
  
# 1.1  
s1 = lambda a, b: a + b  
# 2.1  
s2 = lambda y: math.ceil(y / 100)  
# 3.1  
s3 = lambda s: s[::-1] == s  
# 4.2  
s4 = lambda y: max([a \* b for a, b in zip(y[:-1], y[1:])])  
# 5.2  
s5 = lambda n: (n - 1) \*\* 2 + n \*\* 2  
# 6.2  
s6 = lambda s: max(s) - min(s) - len(set(s)) + 1  
# 7.2\*  
s7 = lambda s: 3 > sum((i >= j) + (i >= k) for i, j, k in zip(s, s[1:], s[2:] + [10 \*\* 6]))  
# 8.2  
s8 = lambda x: sum( x[0] + list(map(sum, [[x[i][j] for j in range(len(x[i])) if x[i - 1][j] != 0] for i in range(1, len(x))])))  
# 9.3  
s9 = lambda y: [i for i in y if len(i) == len(max(y, key=len))]  
# 10.3  
s10 = lambda s1, s2: sum([min(s1.count(i), s2.count(i)) for i in set(s1)])  
# 11.3  
s11 = lambda n: sum(list(map(int, str(n)[:int(len(str(n)) // 2)]))) == sum(list(map(int, str(n)[int(len(str(n)) // 2):])))  
# 12.3  
  
# 13.3\*  
s13 = lambda s: eval('"' + s.replace('(', '"+("').replace(')', '")[::-1]+"') + '"')  
# 14.4  
s14 = lambda a: [sum(a[::2]), sum(a[1::2])]  
# 15.4\*  
s15 = lambda p: ["\*" \* (len(p[0]) + 2)] + ["\*" + i + "\*" for i in p] + ["\*" \* (len(p[0]) + 2)]  
# 16.4\*  
s16 = lambda A, B: sorted(A) == sorted(B) and sum([a != b for a, b in zip(A, B)]) <= 2  
# 17.4  
  
# 18.4  
  
# 19.5  
  
# 20.5\*  
s20 = lambda a: max([abs(a[i]-a[i+1]) for i in range(len(a)-1)])  
# 21.5  
s21 = lambda s: len(s.split('.')) == 4 and all([n.isdigit() and 0 <= int(n) < 256 for n in s.split('.')])  
# 22.5\*  
s22 = lambda ia: min([i for i in range(2, max(ia)+2) if all([j%i!=0 for j in ia])])  
# 23.5\*  
s23 = lambda image: [[int(sum(sum(x[i:i + 3]) for x in image[j:j + 3]) / 9) for i in range(len(image[j]) - 2)] for j in  
 range(len(image) - 2)]  
# 24  
  
# 25  
  
# 26  
  
# 27  
  
# 28  
  
# 29  
  
# 30  
  
# 31  
  
# 32  
  
# 33  
  
# 34  
  
# 35  
  
# 36  
  
# 37  
  
# 38.9\*  
s38 = lambda upSpeed, downSpeed, desiredHeight: 1 if desiredHeight <= upSpeed else (desiredHeight - upSpeed - 1) // ( upSpeed - downSpeed) + 2  
# 39.9\*  
s39 = lambda v1, w1, v2, w2, maxW: max((w1 + w2 <= maxW) \* (v1 + v2), (w1 <= maxW) \* v1, (w2 <= maxW) \* v2)  
# 40.9\*  
s40 = lambda i: re.findall('^\d\*', i)[0]  
# 41.9  
def s41(m, d = 0):  
 while len(str(sum(list(map(int, str(m)))))) > 1: m = str(sum(list(map(int, str(m))))); d =d + 1  
 return d  
# 42.9\*  
s42 = lambda b, p: abs(ord(b[0]) - ord(p[0])) == abs(ord(b[1]) - ord(p[1]))  
# 43.10  
  
# 44  
  
# 45  
  
# 46  
  
# 47  
  
# 48.11  
s48 = lambda s: s.isdigit()  
# 49  
  
# 50  
  
# 51.11\*  
s51 = lambda n: max([int(str(n)[:i] + str(n)[i+1:]) for i in range(len(str(n)))])  
# 52.12\*  
s52 = lambda text: max(re.split('[^a-zA-Z]', text), key=len)  
# 53  
  
# 54  
  
# 55  
  
# 56  
  
# 57  
  
# 58  
  
# 59  
  
# 60  
  
# 61  
  
# 62  
  
# 63

three thousand four hundred fifty-four

import math, re, numpy, pandas  
s1\_1 = lambda a, b: a + b  
s2\_1 = lambda y: math.ceil(y / 100)  
s3\_1 = lambda s: s[::-1] == s  
s4\_1 = lambda y: max([a \* b for a, b in zip(y[:-1], y[1:])])  
s5\_2 = lambda n: (n - 1) \*\* 2 + n \*\* 2  
s6\_2 = lambda s: max(s) - min(s) - len(set(s)) + 1  
s8\_2 = lambda x: sum(x[0] + list(map(sum, [[x[i][j] for j in range(len(x[i])) if x[i - 1][j] != 0] for i in range(1, len(x))])))  
s9\_3 = lambda y: [i for i in y if len(i) == len(max(y, key=len))]  
s10\_3 = lambda s1, s2: sum([min(s1.count(i), s2.count(i)) for i in set(s1)])  
s11\_3 = lambda n: sum(list(map(int, str(n)[:int(len(str(n)) // 2)]))) == sum(list(map(int, str(n)[int(len(str(n)) // 2):])))  
s13\_3 = lambda s: eval('"' + s.replace('(', '"+("').replace(')', '")[::-1]+"') + '"')  
s14\_4 = lambda a: [sum(a[::2]), sum(a[1::2])]  
s21\_5 = lambda s: len(s.split('.')) == 4 and all([n.isdigit() and 0 <= int(n) < 256 for n in s.split('.')])  
s48\_11 = lambda s: s.isdigit()  
s27\_6 = lambda n: n.isidentifier()  
s25\_6 = lambda i, e, s: [x if x != e else s for x in i]  
s26\_6 = lambda n: all(int(d) % 2 == 0 for d in str(n))  
s36\_8 = lambda s: len(set(s))  
  
n7\_2 = lambda s: 3 > sum((i >= j) + (i >= k) for i, j, k in zip(s, s[1:], s[2:] + [10 \*\* 6]))  
n15\_4 = lambda p: ["\*" \* (len(p[0]) + 2)] + ["\*" + i + "\*" for i in p] + ["\*" \* (len(p[0]) + 2)]  
n16\_4 = lambda A, B: sorted(A) == sorted(B) and sum([a != b for a, b in zip(A, B)]) <= 2  
n20\_5 = lambda a: max([abs(a[i] - a[i + 1]) for i in range(len(a) - 1)])  
n22\_5 = lambda ia: min([i for i in range(2, max(ia) + 2) if all([j % i != 0 for j in ia])])  
n23\_5 = lambda image: [[int(sum(sum(x[i:i + 3]) for x in image[j:j + 3]) / 9) for i in range(len(image[j]) - 2)] for j in range(len(image) - 2)]  
n28\_6 = lambda s: "".join(chr((ord(i) - 96) % 26 + 97) for i in s)  
n29\_6 = lambda c1, c2: (ord(c1[0]) + int(c1[1]) + ord(c2[0]) + int(c2[1])) % 2 == 0  
n30\_7 = lambda n, fn: (fn + n / 2) % n  
n31\_7 = lambda d, r, t: math.ceil(math.log(t / d, 1 + r / 100))  
n32\_7 = lambda a: a[(len(a) - 1) // 2]  
n34\_8 = lambda ia, k: [i for (n, i) in enumerate(ia) if (n + 1) % k != 0]  
n38\_9 = lambda u, d, h: 1 if h <= u else (h - u - 1) // (u - d) + 2  
n39\_9 = lambda v1, w1, v2, w2, maxW: max((w1 + w2 <= maxW) \* (v1 + v2), (w1 <= maxW) \* v1, (w2 <= maxW) \* v2)  
n40\_9 = lambda i: re.findall('^\d\*', i)[0]  
n42\_9 = lambda b, p: abs(ord(b[0]) - ord(p[0])) == abs(ord(b[1]) - ord(p[1]))  
n51\_11 = lambda n: max([int(str(n)[:i] + str(n)[i + 1:]) for i in range(len(str(n)))])  
n52\_12 = lambda text: max(re.split('[^a-zA-Z]', text), key=len)  
n33\_7 = lambda w1, w2: sum([a[0] != a[1] for a in zip(w1, w2)]) == 1  
n35\_8 = lambda s: re.search('\\d', s).group(0)  
n18\_4 = lambda i: sum([i.count(i)%2 for i in set(i)]) <= 1  
n19\_5 = lambda ul, ur, fl, fr: {ul, ur} == {fl, fr}  
n44\_10 = lambda a: a[a.rfind("@")+1:]  
n46\_10 = lambda votes, k: (sum(numpy.array(votes) + k > max(votes))) + (k==0 and sum(numpy.array(votes)==max(votes))==1)  
n47\_10 = lambda i: bool(re.fullmatch(r'([A-F\d]{2}-){5}([A-F\d]{2})', i))  
n53\_12 = lambda time: int(time[0:2]) < 24 and int(time[3:]) < 60  
n54\_12 = lambda i: sum(map(int,re.findall('\d+',i)))  
n55\_12 = lambda matrix: len(set(tuple(matrix[o][n] for o in range(i,i+2) for n in range(j,j+2)) for i in range(len(matrix)-1) for j in range(len(matrix[i])-1)))  
n58\_12 = lambda c: "".join([chr(int(c[i:i+8],2)) for i in range(0,len(c),8)])  
  
n12\_3 = 1  
n17\_4 = 1  
n24\_5 = 1  
n37\_8 = 1  
  
n43\_10 = 0  
n45\_10 = 1  
  
n49\_11 = 0  
n50\_11 = 0  
  
n56\_12 = 0  
n57\_12 = 0  
n59\_12 = 0  
n60\_12 = 0  
  
n61\_12 = 0  
n62\_12 = 0  
n63\_12 = 0  
  
def s41\_9(m, d=0):  
 while len(str(sum(list(map(int, str(m)))))) > 1: m = str(sum(list(map(int, str(m))))); d = d + 1  
 return d

abs

all

any

ascii

bin

bool

bytearray

bytes

callable

chr

classmethod

compile

complex

delattr

dict

dir

divmod

enumerate

eval

exec

filter

float

format

frozenset

getattr

globals

hasattr

hash

help

hex

id

input

int

isinstance

issubclass

iter

len

list

locals

map

max

memoryview

min

next

object

oct

open

ord

pow

print

property

range

repr

reversed

round

set

setattr

slice

sorted

staticmethod

str

sum

super

tuple

type

vars

zip

capitalize

casefold

center

count

encode

endswith

expandtabs

find

format

format\_map

index

isalnum

isalpha

isascii

isdecimal

isdigit

isidentifier

islower

isnumeric

isprintable

isspace

istitle

isupper

join

ljust

lower

lstrip

maketrans

partition

replace

rfind

rindex

rjust

rpartition

rsplit

rstrip

split

splitlines

startswith

strip

swapcase

title

translate

upper

zfill

append()

clear()

copy()

count()

extend()

index()

insert()

pop()

remove()

reverse()

sort()

***count()***

clear()

copy()

fromkeys()

get()

items()

keys()

pop()

popitem()

setdefault()

update()

values()

***index()***