

```
1 #-----PROBLEM 1-----
2
3 # Say "Hello, World!" with python
4 print('Hello, World!')
5
6 #Python If-Else
7 import math
8 import os
9 import random
10 import re
11 import sys
12 n = int(input())
13
14 if n % 2 != 0:
15     print('Weird')
16 elif n % 2 == 0 and 2 <= n <= 5:
17     print('Not Weird')
18 elif n % 2 == 0 and 6 <= n <= 20:
19     print('Weird')
20 elif n % 2 == 0 and n > 20:
21     print('Not Weird')
22
23 #Arithmetic Operators
24 a = int(input())
25 b = int(input())
26 print(a+b)
27 print(a-b)
28 print(a*b)
29
30 #Python Division
31 a = int(input())
32 b = int(input())
33 print(a//b)
34 print(a/b)
35
36 #Loops
37 n = int(input())
38 for i in range(n):
39     print(i*i)
40
41 #Write a function
42 def is_leap(year):
43     if year % 400 == 0:
44         return True
45     elif year % 100 == 0:
46         return False
47     elif year % 4 == 0:
48         return True
49     else:
50         return False
51     print(is_leap(year))
```

```

52
53 #Print Function
54 n = int(input())
55 for i in range(1, n+1):
56     print(i, end='')
57
58 #Find the Runner-Up Score!
59 n = int(input())
60 arr = list(map(int, input().split()))
61 first_max = max(arr)
62 while first_max in arr:
63     arr.remove(first_max)
64 second_max = max(arr)
65 print(second_max)
66
67 #Nested List
68 student = []
69 for i in range(int(input())):
70     name = input()
71     score = float(input())
72     student.append([name, score])
73 score = sorted(set([score for name, score in student]))
74 second_lowest_score= score[1]
75 second_lowest_students= sorted([name for name, score in student if score== second_lowest_score])
76 for student in second_lowest_students:
77     print(student)
78
79 #Finding the percentage
80 n= int(input())
81 student_marks = {}
82 for el in range(n):
83     data=input().split()
84     name = data[0]
85     marks = list(map(float, data[1:]))
86     student_marks[name] = marks
87 query_name= input()
88 query_marks= student_marks [query_name]
89 mean = sum(query_marks) / len(query_marks)
90 print(f"{mean:.2f}")
91
92 #Lists
93 if __name__ == '__main__':
94     list= []
95     for _ in range(int(input())):
96         command= input().split()
97         if command[0] == "insert":
98             list.insert(int(command[1]), int(command[2]))
99         elif command[0] == "print":
100             print(list)
101         elif command[0] == "remove":
102             list.remove(int(command[1]))
103         elif command[0] == "append":
104             list.append(int(command[1]))

```

```

105         elif command[0] == "sort":
106             list.sort()
107         elif command[0] == "pop":
108             list.pop()
109         elif command[0] == "reverse":
110             list.reverse()
111
112 #Tuples
113 if __name__ == '__main__':
114     n = int(input())
115     integer_list = map(int, input().split())
116     t = tuple(integer_list)
117     print(hash(t))
118
119 #List Comprehension
120 if __name__ == '__main__':
121     x = int(input())
122     y = int(input())
123     z = int(input())
124     n = int(input())
125     r = [[i, j, k] for i in range (x+1)
126           for j in range (y+1)
127           for k in range (z+1)
128           if i+j+k != n]
129 print(r)
130
131 #String Split and Join
132 def split_and_join(line):
133     a = line.split(" ")
134     return("-".join(a))
135
136 if __name__ == '__main__':
137     line= input()
138     result= split_and_join(line)
139     print(result)
140
141 #What's Your Name?
142 def print_full_name(first, last):
143     print("Hello " + first + " " + last + "! You just delved into python.")
144
145 #Mutations
146 def mutate_string(string, position, character):
147     l=list(string)
148     l[position]= character
149     string= "".join(l)
150     return(string)
151
152 #Find a String
153 def count_substring(string, sub_string):
154     count=0
155     for i in range(len(string)-len(sub_string) + 1):
156         if string[i:i+len(sub_string)] == sub_string:
157             count += 1

```

```

158     return count
159
160 #String Validators
161 if __name__ == '__main__':
162     s = input()
163     print(any(c.isalnum() for c in s))
164     print(any(c.isalpha() for c in s))
165     print(any(c.isdigit() for c in s))
166     print(any(c.islower() for c in s))
167     print(any(c.isupper() for c in s))
168
169 #Text Wrap
170 def wrap(string, max_width):
171     return textwrap.fill(string, max_width)
172
173 #Shape and Reshape
174 import numpy as np
175 my_array = np.array(list(map(int, input().split())))
176 my_array.shape = (3, 3)
177 print(my_array)
178
179 #Capitalize!
180 def solve(s):
181     name = s.split(' ')
182     new_name= []
183     for word in name:
184         if len(word)>0:
185             new_name.append(word[0].upper()+ word[1:])
186         else:
187             new_name.append(word)
188     return' '.join(new_name)
189
190 #Designer Door Mat
191 N, M = map(int,input().split())
192 for i in range(1, N, 2):
193     pattern = ".|." * i
194     print(pattern.center(M, '-'))
195 print("WELCOME".center(M, '-'))
196 for i in range(N-2, 0, -2):
197     pattern = ".|." * i
198     print(pattern.center(M, '-'))
199
200 #Symmetric Difference
201 m = int(input())
202 my_set = set(map(int, input().split()))
203 n = int(input())
204 my_set2 = set(map(int, input().split()))
205 [print(i) for i in sorted(my_set^my_set2)]
206
207 #Set.add()
208 n= int(input())
209 country_name = set()
210 for i in range(n):

```

```
211     country_name.add(input())
212 print(len(country_name))
213
214 #Introduction to sets
215 def average(array):
216     n = set(array)
217     return sum(n) / len(n)
218
219 #Set.union() Operation
220 n = int(input())
221 set_one = set(map(int, input().split()))
222 b= int(input())
223 set_two = set(map(int,input().split()))
224 union = set_one.union(set_two)
225 print(len(union))
226
227 #Set.intersection() Operation
228 n = int(input())
229 set_one = set(map(int,input().split()))
230 b = int(input())
231 set_two = set(map(int,input().split()))
232 inters = set_one.intersection(set_two)
233 print(len(inters))
234
235 #Set.difference() Operation
236 n = int(input())
237 set_one = set(map(int, input().split()))
238 b = int(input())
239 set_two = set(map(int, input().split()))
240 diff = set_one.difference(set_two)
241 print(len(diff))
242
243 #Set.symmetric_difference() Operation
244 n = int(input())
245 set_one = set(map(int, input().split()))
246 b = int(input())
247 set_two = set(map(int, input().split()))
248 symm = set_one ^ set_two
249 print(len(symm))
250
251 #Set Mutations
252 n = int(input())
253 a = set(map(int, input().split()))
254 num = int(input())
255 for i in range(num):
256     m, _ = input().split()
257     new_set = set(map(int, input().split()))
258     if m == "update":
259         a.update(new_set)
260     elif m == "intersection_update":
261         a.intersection_update(new_set)
262     elif m == "difference_update":
263         a.difference_update(new_set)
```

```

264         elif m == "symmetric_difference_update":
265             a.symmetric_difference_update(new_set)
266 print(sum(a))
267
268 #The Captain's Room
269 k = int(input())
270 room = list(map(int,input().split()))
271 a = set(room)
272 cap= (sum(a) * k - sum(room)) // (k-1)
273 print(cap)
274
275 #Check Subset
276 set = int(input())
277 for el in range(test):
278     set_a= int(input())
279     a = set(map(int, input().split()))
280     set_b = int(input())
281     b = set(map(int, input().split()))
282
283     if a <= b:
284         print(True)
285     else:
286         print(False)
287
288 #Check Strict Superset
289 a = set(map(int, input().split()))
290 n = int(input())
291
292 for el in range(n):
293     b = set(map(int, input().split()))
294     if not a.issuperset(b) or a == b:
295         print(False)
296         break
297 else:
298     print(True)
299
300 #No Idea!
301 n, m = map(int, input().split())
302 arr = list(map(int, input().split()))
303 a= set(map(int, input().split()))
304 b= set(map(int, input().split()))
305 happiness = 0
306 for i in arr:
307     if i in a:
308         happiness += 1
309     elif i in b:
310         happiness -= 1
311 print(happiness)
312
313 #The Minion Game
314 def minion_game(string):
315     stuart = 0
316     kevin = 0

```

```

317 V = "AEIOU"
318 n = len(string)
319 for i in range(n):
320     if string[i] in v:
321         kevin += n - i
322     else:
323         stuart += n - i
324 if stuart > kevin:
325     print(f"Stuart {stuart}")
326 elif kevin > stuart:
327     print(f"Kevin {kevin}")
328 else:
329     print("Draw")
330
331 #Calendar Module
332 import calendar
333 month, day, year = map(int, input().split())
334 week_day = calendar.weekday(year, month, day)
335 days= ["MONDAY", "TUESDAY", "WEDNESDAY", "THURSDAY", "FRIDAY", "SATURDAY", "SUNDAY"]
336
337 print(days[week_day])
338
339 #Time Delta
340 import math
341 import os
342 import random
343 import re
344 import sys
345 from datetime import datetime
346
347 def time_delta(t1, t2):
348     fmt = '%a %d %b %Y %H:%M:%S %z'
349     dt1 = datetime.strptime(t1, fmt)
350     dt2 = datetime.strptime(t2, fmt)
351     delta_seconds = abs(int((dt1 - dt2).total_seconds()))
352     return str(delta_seconds)
353
354 #sWAP cASE
355 def swap_case(s):
356     ml= list(s)
357     for i in range(len(ml)):
358         if ml[i].islower():
359             ml[i]= ml[i].upper()
360         elif ml[i].isupper():
361             ml[i]=ml[i].lower()
362     return "".join(ml)
363
364 #String Formatting
365 def print_formatted(number):
366     x= format(number, "b")
367     l= len(x)
368
369     for i in range(1, number+1):
370         octal = format(i, "o")

```

```

370         octal = format(i, "o")
371         hexa = format(i, "X")
372         bina = format(i, "b")
373         print(str(i).rjust(1), str(octal).rjust(1), str(hexa).rjust(1), str(bina).rjust(1))
374
375 #Set.discard().remove() &.pop()
376 n = int(input())
377 s = set(map(int, input().split()))
378 a = int(input())
379 for i in range(a):
380     command = input().split()
381     if command[0] == "pop":
382         s.pop()
383     elif command[0] == "discard":
384         s.discard(int(command[1]))
385     elif command[0] == "remove":
386         s.remove(int(command[1]))
387
388 print(sum(s))
389
390 #Exceptions
391 n = int(input())
392 s = set(map(int, input().split()))
393 a = int(input())
394 for i in range(a):
395     command = input().split()
396     if command[0] == "pop":
397         s.pop()
398     elif command[0] == "discard":
399         s.discard(int(command[1]))
400     elif command[0] == "remove":
401         s.remove(int(command[1]))
402
403 print(sum(s))
404
405 #Transpose and Flatten
406 import numpy as np
407
408 n, m = map(int, input().split())
409 arr = np.array([input().split() for i in range(n)], int)
410 print(np.transpose(arr))
411 print(arr.flatten())
412
413 #Concatenate
414 import numpy as np
415
416 n, m, p = map(int, input().split())
417 arr_1 = np.array([input().split() for _ in range(n)], int)
418 arr_2 = np.array([input().split() for _ in range(m)], int)
419 result = np.concatenate((arr_1, arr_2), axis=0)
420 print(result)
421
422 #Arrays
423 def arrays(arr):

```



```

423 del arrays[arr].
424     arr= numpy.array(arr, float)
425     return arr[::-1]
426
427 #Zeros and Ones
428 import numpy as np
429 arr= tuple(map(int, input().split()))
430 print(np.zeros(arr, dtype=int))
431 print(np.ones(arr, dtype= int))
432
433 #Eye and Identity
434 import numpy
435 numpy.set_printoptions(legacy='1.13')
436 n, m= map(int,input().split())
437 print(numpy.eye(n,m))
438
439 #Sum and Prod
440 import numpy
441 n, m= map(int,input().split())
442 arr = numpy.array([input().split() for _ in range(n)], int)
443 result_s = numpy.sum(arr, axis=0)
444 result_p = numpy.prod(result_s)
445
446 print(result_p)
447
448 #Text Alignment
449 thickness = int(input()) #This must be an odd number
450 c = 'H'
451
452 #Top Cone
453 for i in range(thickness):
454     print((c*i).rjust(thickness-1)+c+(c*i).ljust(thickness-1))
455
456 #Top Pillars
457 for i in range(thickness+1):
458     print((c*thickness).center(thickness*2)+(c*thickness).center(thickness*6))
459
460 #Middle Belt
461 for i in range((thickness+1)//2):
462     print((c*thickness*5).center(thickness*6))
463
464 #Bottom Pillars
465 for i in range(thickness+1):
466     print((c*thickness).center(thickness*2)+(c*thickness).center(thickness*6))
467
468 #Bottom Cone
469 for i in range(thickness):
470     print(((c*(thickness-i-1)).rjust(thickness)+c+(c*(thickness-i-1)).ljust(thickness
471
472 #Array Mathematics
473 import numpy as np
474
475 n, m = map(int, input().split())
476 a = np.array([list(map(int, input().split())) for i in range(n)], int)

```

```

477 b = np.array([list(map(int,input().split())) for i in range(n)], int)
478 print(a + b)
479 print(a - b)
480 print(a * b)
481 print(np.floor_divide(a, b))
482 print(a % b)
483 print(a ** b)
484
485 #Floor, Ceil and Rint
486 import numpy as np
487 np.set_printoptions(legacy= '1.13')
488 a = np.array(list(map(float, input().split()))))
489
490 print(np.floor(a))
491 print(np.ceil(a))
492 print(np rint(a))
493
494 #Min and Max
495 import numpy as np
496 n,m = map(int, input().split())
497 my_arr = np.array([list(map(int, input().split())) for i in range(n)])
498
499 minimo = np.min(my_arr, axis= 1)
500 massimo = np.max (minimo)
501
502 print(massimo)
503
504 #Mean, var and Std
505 import numpy as np
506 n, m = map(int, input().split())
507 my_arr = np.array([(list(map(int, input().split())))) for i in range(n)])
508 print(np.mean(my_arr, axis=1))
509 print(np.var (my_arr, axis = 0))
510 print(np.round(np.std(my_arr, axis=None), 11))
511
512 #Dot and Cross
513 import numpy as np
514 n =int(input())
515
516 a = np.array([list(map(int,input().split())) for i in range(n)])
517 b = np.array([list(map(int,input().split())) for i in range(n)])
518
519 dot= np.dot(a,b)
520 print(dot)
521
522 #Inner and Outer
523 import numpy as np
524
525 a = np.array(list(map(int, input().split()))))
526 b = np.array(list(map(int, input().split()))))
527
528 print(np.inner(a, b ))
529 print(np.outer(a ,b ))

```

```
530
531 #Polynomials
532 import numpy as np
533
534 c = list(map(float, input().split()))
535 x = float(input())
536
537 value = np.polyval(c, x)
538 print(value)
539
540 #Linear Algebra
541 import numpy as np
542 n= int(input())
543 a= np.array([list(map(float, input().split())) for i in range(n)])
544 det= np.linalg.det(a)
545 print(np.round(det ,2))
546
547 #Map and Lamba Function
548 cube = lambda x: x**3
549
550 def fibonacci(n):
551     if n==0:
552         return []
553     elif n==1:
554         return [0]
555     elif n==2:
556         return [0,1]
557     else:
558         result = fibonacci(n-1)
559         result.append(result[-1] + result[-2])
560     return result
561
562 #Zipped!
563 n, x = map(int, input().split())
564 my_l = []
565 for i in range(x):
566     my_l.append(list(map(float, input().split())))
567 for m in zip(*my_l):
568     print(sum(m)/len(m))
569
570 #Athlete Sort
571 import math
572 import os
573 import random
574 import re
575 import sys
576
577
578
579 if __name__ == '__main__':
580     nm = input().split()
581
582     n = int(nm[0])
```

```
583
584     m = int(nm[1])
585
586     arr = []
587
588     for _ in range(n):
589         arr.append(list(map(int, input().rstrip().split()))))
590
591     k = int(input())
592     arr.sort(key=lambda x: x[k])
593     for row in arr:
594         print(*row)
595
596 #Collections.counter
597 from collections import Counter
598 input()
599 i = Counter(map(int, input().split()))
600 c = int(input())
601 p = 0
602
603 for _ in range(c):
604     t, z = map(int, input().split())
605     if i[t] > 0:
606         p += z
607         i[t] -= 1
608
609 print(p)
610
611 #Collections.namedtuple()
612 from collections import namedtuple
613 n = int(input())
614 columns = input().split()
615 students = namedtuple('students', columns)
616 tm = []
617 for i in range(n):
618     student = students(*input().split())
619     tm.append(int(student.MARKS))
620
621 print(round(sum(tm)/len(tm), 2))
622
623 #DefaultDict Tutorial
624 from collections import defaultdict
625 n, m = map(int, input().split())
626 A= defaultdict(list)
627 for i in range(n):
628     w = input()
629     A[w].append(i+1)
630 for _ in range(m):
631     w= input()
632     if w in A:
633         print(" ".join(map(str, A[w])))
634     else:
635         print(-1)
```

```
636
637 #Collections.OrderedDict()
638 from collections import OrderedDict
639 dic = OrderedDict()
640 n = int(input())
641 for i in range(n):
642     *item_name, price = input().split()
643     item_name = " ".join(item_name)
644     price = int(price)
645
646     if item_name in dic:
647         dic[item_name] += price
648     else:
649         dic[item_name] = price
650
651 for item, price in dic.items():
652     print(item, price)
653
654 #Collections.deque()
655 from collections import deque
656 d = deque()
657 n = int(input())
658 for i in range(n):
659     o = input().split()
660     c = o[0]
661
662     if c == "append":
663         d.append(int(o[1]))
664     if c == "pop":
665         d.pop()
666     if c == "appendleft":
667         d.appendleft(int(o[1]))
668     if c == "popleft":
669         d.popleft()
670
671 print(" ".join(map(str, d)))
672
673 #Word Order
674 n = int(input())
675 dic = {}
676
677 for i in range(n):
678     w = input()
679     if w in dic:
680         dic[w] += 1
681     else:
682         dic[w] = 1
683
684 print(len(dic))
685 print(*dic.values())
686
687 #Company Logo
688 import math
```

```

689 import os
690 import random
691 import re
692 import sys
693 from collections import Counter
694
695
696 if __name__ == '__main__':
697     s = input().strip()
698     counter = Counter(s)
699     sorted_counter = sorted(counter.items(), key=lambda x: (-x[1], x[0]))
700
701     for i in range(3):
702         print(f"{sorted_counter[i][0]} {sorted_counter[i][1]}")
703         # i looked the solutions
704
705 #Piling Up!
706 from collections import deque
707
708 for _ in range(int(input())):
709     n = int(input())
710     blocks = deque(map(int, input().split()))
711     possible = True
712     last = max(blocks[0], blocks[-1])
713     while blocks:
714         if blocks[0] >= blocks[-1]:
715             current = blocks.popleft()
716         else:
717             current = blocks.pop()
718         if current > last:
719             possible = False
720             break
721         last = current
722     print("Yes" if possible else "No")
723
724     #i looked the solutions
725
726 #Alphabet Rangoli
727 def print_rangoli(size):
728     import string
729     alpha = string.ascii_lowercase
730     lines = []
731     for i in range(size):
732         s = "-".join(alpha[size-1-i:-1] + alpha[i:size])
733         lines.append(s.center(4 * size - 3, '-'))
734     print('\n'.join(lines[::-1] + lines[1:]))
735
736 #Decorators 2- Name directory
737 def person_lister(f):
738     def inner(people):
739         return [f(person) for person in sorted(people, key=lambda person: int(person[1]))]
740     return inner
741

```

```

742 # i looked the solutions
743
744 #Merge the tools!
745 def merge_the_tools(string, k):
746     for i in range(0, len(string), k):
747         substring = string[i:i+k]
748         result = ""
749         for char in substring:
750             if char not in result:
751                 result += char
752         print(result)
753
754 #i looked the solutions
755
756 #Standardize Mobile Number Using Decorators
757 def wrapper(f):
758     def fun(l):
759         l = ['+91 ' + p[-10:-5] + ' ' + p[-5:] for p in [num.strip()[-10:] for num in l]]
760         f(l)
761     return fun
762 #i looked the solutions
763
764 #Detect Floating point Number
765 t = int(input())
766 for i in range(t):
767     n = input().strip()
768     if n.startswith(('+', '-')) or '.' in n:
769         try:
770             if float(n):
771                 print(True)
772         except ValueError:
773             print(False)
774     else:
775         print(False)
776
777 #ginortS
778 def order_letters(s):
779     lower = []
780     upper = []
781     odd = []
782     even = []
783
784     for i in s:
785         if i.islower():
786             lower.append(i)
787         elif i.isupper():
788             upper.append(i)
789         elif i.isdigit():
790             if int(i) % 2 != 0:
791                 odd.append(i)
792             else:
793                 even.append(i)
794

```

```

795
796     lower.sort()
797     upper.sort()
798     odd.sort()
799     even.sort()
800
801
802     result = ''.join(lower + upper + odd + even)
803
804
805     return result
806
807 if __name__ == "__main__":
808     s = input()
809     print(order_letters(s))
810
811
812 #-----PROBLEM 2-----
813 #Birthday Cake Candles
814 import math
815 import os
816 import random
817 import re
818 import sys
819
820 def birthdayCakeCandles(candles):
821     max_height = max(candles)
822     count = candles.count(max_height)
823     return count
824
825 if __name__ == '__main__':
826     fptr = open(os.environ['OUTPUT_PATH'], 'w')
827
828     candles_count = int(input().strip())
829
830     candles = list(map(int, input().rstrip().split()))
831
832     result = birthdayCakeCandles(candles)
833
834     fptr.write(str(result) + '\n')
835
836     fptr.close()
837
838 #Number Line Jump
839 import math
840 import os
841 import random
842 import re
843 import sys
844
845 def kangaroo(x1, v1, x2, v2):
846
847     if v1==v2:

```



```

848         if x1 == x2:
849             return "YES"
850         else:
851             return "NO"
852     if (x2 - x1) % (v1-v2) == 0 and (x2-x1) / (v1 - v2) > 0:
853         return "YES"
854     else:
855         return "NO"
856
857
858 if __name__ == '__main__':
859     fptr = open(os.environ['OUTPUT_PATH'], 'w')
860
861     first_multiple_input = input().rstrip().split()
862
863     x1 = int(first_multiple_input[0])
864
865     v1 = int(first_multiple_input[1])
866
867     x2 = int(first_multiple_input[2])
868
869     v2 = int(first_multiple_input[3])
870
871     result = kangaroo(x1, v1, x2, v2)
872
873     fptr.write(result + '\n')
874
875     fptr.close()
876
877
878 print(kangaroo(x1, v1, x2, v2))
879
880 #Viral Advertising
881 import math
882 import os
883 import random
884 import re
885 import sys
886
887 def viralAdvertising(n):
888     p = 5
889     c = 0
890     like_today=0
891
892     for i in range(n):
893         like_today = p//2
894         c += like_today
895         p = like_today*3
896     return c
897
898 if __name__ == '__main__':
899     fptr = open(os.environ['OUTPUT_PATH'], 'w')
900
901     n = int(input().rstrip())

```

```

901 n = int(input().strip())
902
903 result = viralAdvertising(n)
904
905 fptr.write(str(result) + '\n')
906
907 fptr.close()
908
909 #Recursive Digit Sum
910 def superDigit(n, k):
911     c = str(sum(int(i) for i in str(n)))*k
912     t=0
913     a=0
914     for i in c:
915         t += int(i)
916     while len(str(t))>1:
917         a=0
918         for i in str(t):
919             a+= int(i)
920         t=a
921     return a
922
923 if __name__ == '__main__':
924     fptr = open(os.environ['OUTPUT_PATH'], 'w')
925
926     first_multiple_input = input().rstrip().split()
927
928     n = first_multiple_input[0]
929
930     k = int(first_multiple_input[1])
931
932     result = superDigit(n, k)
933
934     fptr.write(str(result) + '\n')
935
936     fptr.close()
937
938 #Inserion Sort 1
939 import math
940 import os
941 import random
942 import re
943 import sys
944
945 def insertionSort1(n, arr):
946     e = arr[-1]
947     for i in range(n-2, -1, -1):
948         if arr[i] > e:
949             arr[i+1] = arr[i]
950             print(*arr)
951         else:
952             arr[i+1] = e
953             print(*arr)
954             break

```

```

954         break
955     else:
956         arr[0]= e
957         print(*arr)
958
959 if __name__ == '__main__':
960     n = int(input().strip())
961
962     arr = list(map(int, input().rstrip().split()))
963
964     insertionSort1(n, arr)
965
966 #Insertion Sort 2
967 import math
968 import os
969 import random
970 import re
971 import sys
972
973 def insertionSort2(n, arr):
974     for i in range(1, n):
975         k = arr[i]
976         for j in range(i - 1, -2, -1):
977             if j >= 0 and arr[j] > k:
978                 arr[j + 1] = arr[j]
979             else:
980                 arr[j + 1] = k
981                 break
982         print(*arr)
983
984 if __name__ == '__main__':
985     n = int(input().strip())
986
987     arr = list(map(int, input().rstrip().split()))
988
989     insertionSort2(n, arr)
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007

```

1008
1009
1010



Mostra output nascosti

Nuova sezione