*"""  
This is a question in the very hard category. It actually involves the use of  
v - e + r = 2 for calculating the number of regions generated by overlapping  
polygons by finding number of points of intersection.  
  
The polygons are separated into sets of intersecting polygons and the above  
formula is applied to every such set. This division is done by creating a  
relational graph where the polygons are represented by nodes and the relation  
being their intersection. Then each set is a connected component of this graph.  
  
Programming has been simplified to directly finding number of intersections and  
finding number of connected components.  
  
Knowledge of geometry is rigorously tested in this question.  
"""*n = int(input()) *# number of polygons*poly\_graph = [[0 **for** \_ **in** range(n)] **for** \_ **in** range(n)] *# graph of polygons as an adjacency matrix*done\_poly = [i **for** i **in** range(n)] *# array of polygons which have not yet been considered in any set*n\_edges = 0 *# number of edges*poly = list() *# list of all vertices separated into polygons  
  
# input of all polygons***for** i **in** range(n):  
 p = list()  
 p\_count = int(input())  
 **for** j **in** range(p\_count):  
 v = [float(i) **for** i **in** input().split(**' '**)]  
 p.append(v)  
 n\_edges += len(p)  
 poly.append(p)  
  
n\_vertices = n\_edges *# number of vertices is same as number of edges in a polygon (intersections not considered yet)***def** intersects(l1, l2):  
 *# a functions that checks whether 2 edges l1 and l2 are intersecting or not* a1 = l1[1][1] - l1[0][1]  
 b1 = l1[0][0] - l1[1][0]  
 c1 = l1[1][0] \* l1[0][1] - l1[0][0] \* l1[1][1]  
 a2 = l2[1][1] - l2[0][1]  
 b2 = l2[0][0] - l2[1][0]  
 c2 = l2[1][0] \* l2[0][1] - l2[0][0] \* l2[1][1]  
 **if** b1 **is** 0 **and** b2 **is** 0 **or** a1 **is** 0 **and** a2 **is** 0:  
 **return False  
 if** a1 \* b2 == a2 \* b1:  
 **return False** Y = (a1 \* c2 - a2 \* c1) / (a2 \* b1 - a1 \* b2)  
 **if not** a1 == 0:  
 X = -(c1 + b1 \* Y) / a1  
 **else**:  
 X = -(c2 + b2 \* Y) / a2  
 **if** X >= min(l1[0][0], l1[1][0]) **and** \  
 X <= max(l1[0][0], l1[1][0]) **and** \  
 X >= min(l2[0][0], l2[1][0]) **and** \  
 X <= max(l2[0][0], l2[1][0]) **and** \  
 Y >= min(l1[0][1], l1[1][1]) **and** \  
 Y <= max(l1[0][1], l1[1][1]) **and** \  
 Y >= min(l2[0][1], l2[1][1]) **and** \  
 Y <= max(l2[0][1], l2[1][1]):  
 **return True  
 return False**intersections = 0 *# stores number of intersections with edges of polygon considered***for** i **in** range(len(poly)): *# iterating through polygons* **for** j **in** range(i + 1, len(poly)): *# iterating through polygons not considered for intersection with poly[i]* **for** k **in** range(len(poly[i])): *# iterating through edges of poly[i]* intersections = 0 *# initialising number of intersections* **for** l **in** range(len(poly[j])): *# iterating through edges in poly[j]* **if** intersects([poly[i][k], poly[i][(k + 1) % len(poly[i])]],  
 [poly[j][l], poly[j][(l + 1) % len(poly[j])]]):  
 intersections += 1 *# counting intersections* n\_vertices += intersections *# every intersection creates a new vertex* n\_edges += 2 \* intersections *# every intersection creates two new edges* **if** intersections >= 1: *# if there exists even a single intersection then the two polygons overlap* poly\_graph[i][j], poly\_graph[j][i] = 1, 1 *# creating the overlap edge in relational polygon*overlap\_sets = 0 *# counts number of overlapping sets***def** remove\_poly(s):  
 *# removes all polygons that belong to the same set recursively* **if** s **not in** done\_poly:  
 **return** done\_poly.remove(s)  
 **for** i **in** range(len(poly\_graph[s])):  
 **if** poly\_graph[s][i] == 1:  
 remove\_poly(i)  
  
  
**while** len(done\_poly) **is not** 0:  
 remove\_poly(done\_poly[0])  
 overlap\_sets += 1  
  
regions = 2 + n\_edges - n\_vertices + overlap\_sets - 1  
*# CALCULATING THE NUMBER OF REGIONS USING THE FORMULA V - E + R = 2*print(regions \* 5) *# output (5 samples per region)*