*"""  
This is an easy category question based on the principal of MST(Minimum Spanning Tree)  
  
Take the input vertices and create a fully connected graph from those vertices with  
undirected edges having weights equal to distance of separation. The sum of weights of  
the minimum spanning tree is the solution.  
"""***from** math **import** hypot  
**from** queue **import** PriorityQueue  
  
n = int(input())  
coords = [[float(i) **for** i **in** input().split()[1:]] **for** j **in** range(n)] *# input coordinates*dist = [[0 **for** \_ **in** range(len(coords))] **for** \_ **in** range(len(coords))] *# distance matrix*pq = PriorityQueue(n \* n)  
  
*# matrix implementation of the graph***for** i **in** range(n):  
 **for** j **in** range(n):  
 dist[i][j] = hypot(coords[i][0] - coords[j][0], coords[i][1] - coords[j][1])  
  
*# using unoptimised Prim's Algorithm*mind = 0  
done = [0 **for** \_ **in** range(n)]  
done[0] = 1  
**for** i **in** range(1, n):  
 pq.put([dist[0][i], 0, i])  
  
**while** n > 0 **and not** pq.empty():  
 e = pq.get()  
 **if** done[e[2]] **is not** 1:  
 mind += e[0]  
 n -= 1  
 done[e[2]] = 1  
 **for** i **in** range(len(coords)):  
 **if** i **is not** e[2] **and** done[i] **is not** 1:  
 pq.put([dist[e[2]][i], e[2], i])  
  
print(round(mind \* 10))