

  
def generate\_permutations(lst): # Генерирует все возможные перестановки заданного списка.  
  
 # Бұл шарт lst тізімінің ұзындығы 1-ден кіші немесе тең екенін тексереді.  
 # Если условие истинно то функция возвращает список, содержащий сам lst  
 if len(lst) <= 1:  
 return [lst]  
# result = []: Создается пустой список result, который будет содержать все перестановки.  
# for i in range(len(lst)):: for циклі lst тізімінің элементтерінің индекстері арқылы өтеді.  
# current\_element = lst[i]: Текущий элемент i индексіндегі тізім элементіне орнатылған.  
# `remaining\_elements = lst[:i] + lst[i + 1:]` өрнегі 'lst' бастап 'i' индексіне (қоса) дейінгі барл эл  
# 'remaining\_elements' жаңа тізімін жасайды `.  
 result = []  
 for i in range(len(lst)):  
 current\_element = lst[i]  
 remaining\_elements = lst[:i] + lst[i + 1:]  
  
 # Рекурсивно генерируем перестановки для оставшихся элементов(қалған элементтерге рек ген)  
 permutations\_of\_remaining = generate\_permutations(remaining\_elements)  
  
 # Собираем текущую перестановку, добавляя текущий элемент  
 for perm in permutations\_of\_remaining:  
 result.append([current\_element] + perm)  
 return result  
# Пример использования функции  
my\_list = [1, 2, 3]  
permutations = generate\_permutations(my\_list)  
  
# Выводим все перестановки Әрбір перестановка үшін текущий элементтен басталатын жаңа тізім жасалады,  
for perm in permutations:  
 print(perm)

