Задача 1.

Лёгкая

**Сортировка пузырьком: количество обменов**

**Ограничение по времени работы программы: 1 секунда**

Дан список целых чисел. Какое количество обменов совершит пузырьковая сортировка (по неубыванию элементов), запущенная на этом списке?

ВХОДНЫЕ ДАННЫЕ

Программа получает на вход n целых чисел, записанных через пробел в одной строке, 1⩽n⩽100. Сами числа по модулю не превосходят 109

ВЫХОДНЫЕ ДАННЫЕ

Программа должна вывести одно число — искомое количество обменов.

ПРИМЕР

|  |  |
| --- | --- |
| **ввод** | **вывод** |
| 6 2 4 1 | 5 |

Начало формы

Задача 2.

Вводная

**Медиана списка**

**Ограничение по времени работы программы: 1 секунда**

Дан список целых чисел. Найдите в нем “медианный” элемент, то есть то число, которое будет ровно посередине списка, если список отсортировать.

ВХОДНЫЕ ДАННЫЕ

Программа получает на вход n целых чисел, записанных через пробел в одной строке, 1⩽n⩽100, значение n — нечетное. Сами числа по модулю не превосходят 109

ВЫХОДНЫЕ ДАННЫЕ

Программа должна вывести одно число — медианный элемент данного списка.

ПРИМЕР

|  |  |
| --- | --- |
| **ввод** | **вывод** |
| 2 8 1 5 4 | 4 |

Начало формы

Задача 3.

Лёгкая

**Создание архива**

**Ограничение по времени работы программы: 1 секунда**

Системный администратор вспомнил, что давно не делал архива пользовательских файлов. Однако объем диска, куда он может поместить архив, может быть меньше, чем суммарный объем архивируемых файлов.

Известно, какой объем занимают файлы каждого пользователя.

Напишите программу, которая по заданной информации о пользователях и свободному объему на архивном диске определит максимальное число пользователей, чьи данные можно поместить в архив, при этом используя свободное место как можно более полно.

ВХОДНЫЕ ДАННЫЕ

Программа получает на вход в одной строке число S – размер свободного места на диске (натуральное, не превышает 10000), и число N – количество пользователей (натуральное, не превышает 100), после этого идет N чисел - объем данных каждого пользователя (натуральное, не превышает 1000), записанных каждое в отдельной строке.

ВЫХОДНЫЕ ДАННЫЕ

Выведите наибольшее количество пользователей, чьи данные могут быть помещены в архив.

ПРИМЕР

|  |  |
| --- | --- |
| **ввод** | **вывод** |
| 100 2 200 50 | 1 |
| 100 3 50 30 50 | 2 |

Начало формы

 Решение задачи №1

Модифицируем алгоритм пузырьковой сортировки так, чтобы он подсчитывал количество выполненных обменов. Заведем внутри алгоритма переменную count, которую будем увеличивать на 1 при выполнении обмена. В конце вернем значение count.

def BubbleSort(A):  
    count = 0  
    for j in range(len(A) - 1, 0, -1):  
        for i in range(0, j):  
            if A[i] > A[i + 1]:  
                A[i], A[i + 1] = A[i + 1], A[i]  
                count += 1  
    return count  
  
A = list(map(int, input().split()))  
print(BubbleSort(A))

Решение задачи №2

Считаем файл, создадим список из считанных чисел. Отсортируем его при помощи любого из алгоритмов сортировки. Выведем элемент с индексом n//2.

def InsertionSort(A):  
    for i in range(1, len(A)):  
        new\_elem = A[i]  
        j = i - 1  
        while j >= 0 and A[j] > new\_elem:  
            A[j + 1] = A[j]  
            j -= 1  
        A[j + 1] = new\_elem  
  
A = list(map(int, input().split()))  
InsertionSort(A)  
print(A[len(A) // 2])

Решение задачи №3

Отсортируем список размеров данных пользователей. Нужно брать минимальные элементы списка. Одновременно с выбором элемента уменьшаем значение S (свободного места для архивирования) на значение выбранного элемента. Цикл продолжается, пока значение выбираемого элемента не превосходит S.

РЕШЕНИЕ НА ЯЗЫКЕ PYTHON

def InsertionSort(A):  
    for i in range(1, len(A)):  
        new\_elem = A[i]  
        j = i - 1  
        while j >= 0 and A[j] > new\_elem:  
            A[j + 1] = A[j]  
            j -= 1  
        A[j + 1] = new\_elem  
  
S, N = map(int, input().split())  
A = []  
for i in range(N):  
    A.append(int(input()))  
InsertionSort(A)  
i = 0  
while i < len(A) and A[i] <= S:  
    S -= A[i]  
    i += 1  
print(i)

Конец формы

Конец формы

Конец формы