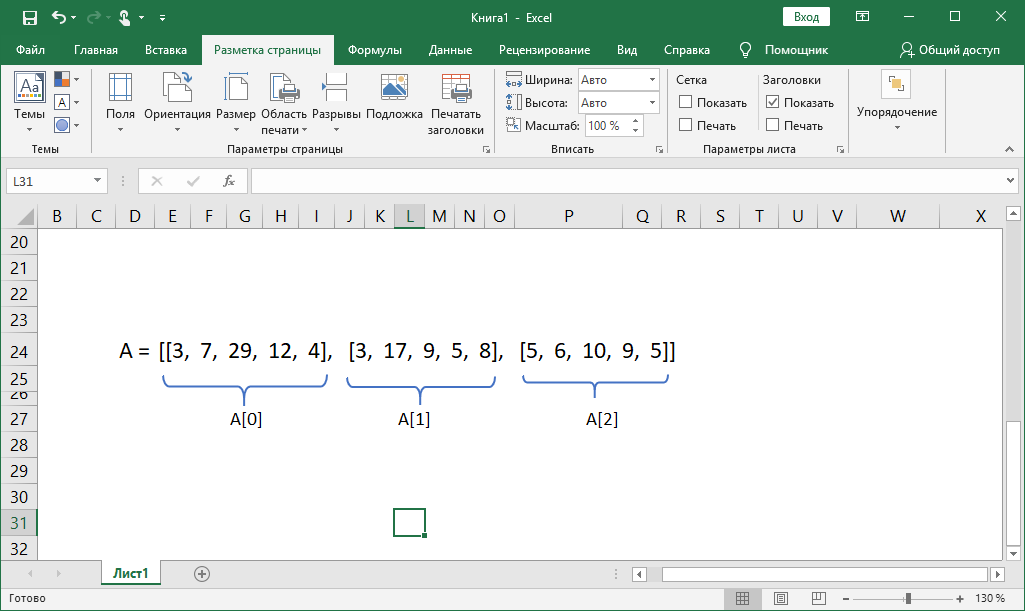
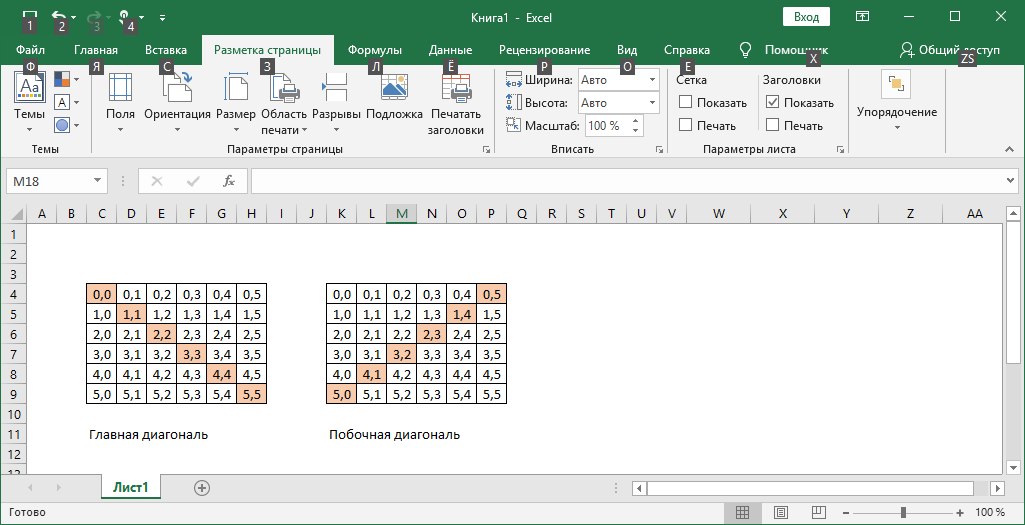
Двумерные массивы (вложенные списки)

В языке программирования Python таблицу можно представить в виде списка, каждый элемент которого является в свою очередь списком, например, чисел:



Индексы двумерного массива:



|  |  |
| --- | --- |
| **Формулировка задачи** | **Код** |
| **Ввод двумерного массива** | |
| На вход программы подается n, m – количество строк и столбцов, затем сам массив:  3 5  3 4 5 6 3  3 4 5 2 1  1 5 3 2 8 | n, m = map(int, input().split())  A = [ ] # создаём пустой одномерный массив A  for i in range(n):  A.append (list (map (int, input().split()))) # считываем n одномерных массивов  # и добавляем в исходный массив А |
| Предыдущая задача. При считывании массива используем генератор. | A = [list(map(int, input().split())) for i in range(n)] |
| Задача та же, но без использования сложных вызовов функций. | n, m = map(int, input().split())  A = [ ] # создаём пустой одномерный массив A  for i in range(n):  row = input().split() # считываем строку и разбиваем по пробелам  for i in range (len (row) ):  row[i] = int(row[i]) # каждый элемент считанного массива переводим в int  A.append(row) # полученный числовой массив добавляем в  # исходный массив А |
| **Создание двумерного массива, состоящего из 0** | |
| Даны два числа: количество строк n и количество столбцов m. Необходимо создать список размером n\*m, заполненный нулями. | A = [0] \* n # создаём нулевой одномерный массив  for i in range (n):  A[i] = [0] \* m # каждому элементу исходного списка присваиваем  # массив из m элементов |
| Задача та же | A = [ ] # создаём пустой одномерный массив  for i in range (n):  A.append ([0] \* m) # добавляем в исходный массив одномерные массивы,  # состоящие из 0 |
| Создать двумерный нулевой массив с помощью генератора | A = [ [0]\*m for i in range(n) ] |
| **Обработка двумерного массива** | |
| Дан массив n\*n. Необходимо найти сумму элементов, стоящих выше главной диагонали. | S = 0  for i in range (n): # перебираем строки  for j in range (n): # перебираем элементы массива в каждой строке  if j > i: # у элементов выше главной диагонали индекс j > индекса i  S = S + 1  print(S) # по окончании работы вложенных циклов, выводим S |
| Дан массив n\*m. Необходимо найти сумму элементов каждой строки. | for i in range (n): # перебираем строки  S = 0  for j in range (m): # перебираем элементы массива в каждой строке  S = S + A[i][j] # прибавляем к S очередной элемент массива  print(S) # по окончании работы внутреннего цикла, выводим S |
| Найти сумму элементов в массиве, организовав перебор самих элементов, не затрагивая индексы. | S = 0  for row in A: # перебираем строки  for elem in row: # перебираем элементы в строке  S += elem # прибавляем очередной элемент массива |
| **Вывод двумерного массива** | |
| Вывести двумерный числовой список n\* m на экран построчно, разделяя числа пробелами внутри одной строки | for i in range (n): # перебираем строки  for j in range (m): # перебираем элементы строки  print (A[i][j], end=" ") # выводим элемент строки, затем пробел  print () # по окончании вывода строки, переводим курсор  # на новую строку |
| Вывести двумерный числовой список на экран построчно, разделяя числа пробелами внутри одной строки | for i in range (len(A)): # перебираем строки в количестве len(A)  for j in range (len(A[i]): # перебираем элементы строки в количестве len(A[i])  print (A[i][j], end=" ") # выводим элемент строки, затем пробел  print () # по окончании вывода строки, переводим курсор  # на новую строку |
| Вывести двумерный список, организовав переборы в циклах не по индексу, а по значениям списка | for row in A:  for elem in row:  print (elem, end = " ")  print () |
| Вывести двумерный список, организовав вывод строк с помощью метода join. | for row in A:  print (" ".join (list (map (str, row)))) |