Теоретический материал по языку программирования Python. Двумерные массивы.

Гурин Семен

18 января 2020

1 Введение

Часто в задачах приходится хранить прямоугольные таблицы с данными. Такие таблицы называются матрицами или двумерными массивами. В языке программирования Питон таблицу можно представить в виде списка строк, каждый элемент которого является в свою очередь списком, например, чисел. Например, создать числовую таблицу из двух строк и трех столбцов можно так:

$$A = [[1, 2, 3], [4, 5, 6]]$$

По сути, таким образом представляется такая таблица (матрица):

$$A = \begin{array}{c} \\ 0 \\ 1 \end{array} \begin{array}{c} \begin{array}{ccc} 0 & 1 & 2 \end{array} \\ \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix} \end{array}$$

Здесь первая строка списка A[0] является списком из чисел [1, 2, 3]. То есть:

$$A[0][0] == 1$$
$$A[0][1] == 2$$
$$A[0][2] == 3$$
$$A[1][0] == 4$$
$$A[1][1] == 5$$
$$A[1][2] == 6$$

2 Обработка и вывод двумерного массива

Для обработки и вывода списка как правило используется два вложенных цикла. Первый цикл по номеру строки, второй цикл по элементам внутри строки. Например, вывести двумерный числовой список на экран построчно, разделяя числа пробелами внутри одной строки, можно так:

```
for i in range(len(A)):
    for j in range(len(A[i]):
        print(A[i][j], end = ' ')
    print()
```

То же самое, но циклы не по индексу, а по значениям списка:

```
for row in A:
    for elem in row:
        print(elem, end = ' ')
    print()
```

Используем два вложенных цикла для подсчета суммы всех чисел в списке:

```
S = 0
for i in range(len(A)):
    for j in range(len(A[i])):
        S += A[i][j]
```

Или то же самое с циклом не по индексу, а по значениям строк:

```
S = 0
for row in A:
    for elem in row:
        S += elem
```

3 Создание вложенных списков

Пусть даны два числа: количество строк n и количество столбцов m. Необходимо создать список размером $n \times m$, заполненный нулями.

3.1 Способ I

Сначала создадим список из n элементов (для начала просто из n нулей). Затем сделаем каждый элемент списка ссылкой на другой одномерный список из m элементов:

```
A = [0] * n
for i in range(n):
    A[i] = [0] * m
```

3.2 Способ II

Другой (но похожий) способ: создать пустой список, потом n раз добавить в него новый элемент, являющийся списком-строкой:

```
A = []
for i in range(n):
    A.append([0] * m)
```

3.3 Способ III

Но еще проще воспользоваться генератором: создать список из n элементов, каждый из которых будет списком, состоящих из m нулей:

```
A = [ [0] * m for i in range(n)]
```

4 Пример обработки двумерного массива

Пусть дан квадратный массив из n строк и n столбцов. Необходимо элементам, находящимся на главной диагонали, проходящей из левого верхнего угла в правый нижний (то есть тем элементам A[i][j], для которых i==j) присвоить значение 1, элементам, находящимся выше главной диагонали – значение 0, элементам, находящимся ниже главной диагонали – значение 2. То есть получить такой массив (пример для n==4):

1	0	0	0
2	1	0	0
2	2	1	0
2	2	2	1

Рассмотрим несколько способов решения этой задачи. Элементы, которые лежат выше главной диагонали – это элементы A[i][j], для которых i<j, а для элементов ниже главной диагонали i>j. Таким образом, мы можем сравнивать значения i и j и по ним определять значение A[i][j]. Получаем следующий алгоритм:

```
for i in range(n):
    for j in range(n):
        if i < j:
            A[i][j] = 0
        elif i > j:
            A[i][j] = 2
        else:
            A[i][j] = 1
```

Данный алгоритм плох, поскольку выполняет одну или две инструкции if для обработки каждого элемента. Если мы усложним алгоритм, то мы сможем обойтись вообще без условных инструкций.
Сначала заполним главную диагональ, для чего нам понадобится один цикл:

```
for i in range(n):
    A[i][i] = 1
```

Затем заполним значением 0 все элементы выше главной диагонали, для чего нам понадобится в каждой из строк с номером i присвоить значение элементам A[i][j] для $j = i + 1, ..., n - 1$. Здесь нам понадобятся вложенные циклы:

```
for i in range(n):
    for j in range(i + 1, n):
        A[i][j] = 0
```

Аналогично присваиваем значение 2 элементам A[i][j] для $j = 0, ..., i - 1$:

```
for i in range(n):
    for j in range(0, i):
        A[i][j] = 2
```

Можно также внешние циклы объединить в один и получить еще одно, более компактное решение:

```
for i in range(n):
    for j in range(0, i):
        A[i][j] = 2
    A[i][i] = 1
    for j in range(i + 1, n):
        A[i][j] = 0
```

5 Ввод двумерных массивов

Пусть программа получает на вход двумерный массив, в виде n строк, каждая из которых содержит m чисел, разделенных пробелами. Как их считать? Например, так:

```
A = []
for i in range(n):
    A.append(list(map(int, input().split())))
```

Или, без использования сложных вложенных вызовов функций:

```
A = []
for i in range(n):
    row = input().split()
    for i in range(len(row)):
        row[i] = int(row[i])
    A.append(row)
```

Можно сделать то же самое и при помощи генератора:

```
A = [ list(map(int, input().split())) for i in range(n)]
```