Массивы и строки в языке Python

Массив – совокупность пронумерованных величин одного типа, объединённых общим именем.

В языке Python нет такой структуры данных, как массив.

Для хранения группы однотипных объектов используют ***списки*** (тип данных **`list`**).

Индекс – порядковый номер элемента в массиве.

Нумерация элементов массива всегда начинается с нуля.

Каждый элемент массива обозначается *индексированным именем*:

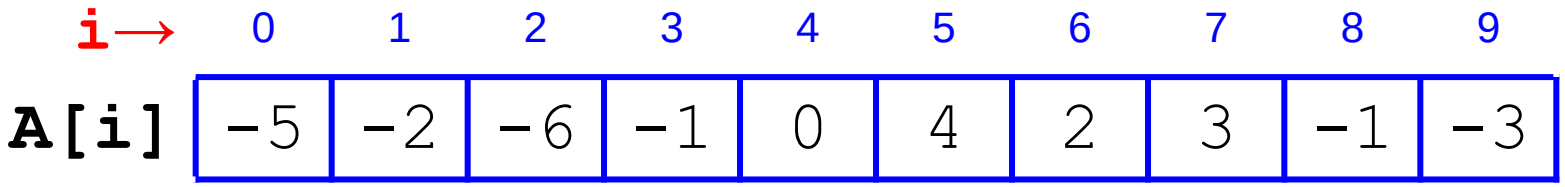
`Имя[индекс]`

Например:

`A[1]` – второй элемент массива A (с индексом 1).

Массивы бывают *одномерные* (*линейные*) и *двумерные* (*прямоугольные*). Далее рассматриваются одномерные массивы.

Одномерный (линейный) массив **A**



A – *имя массива*

i – *индекс элемента*

A[0] = -5, A[1] = -2, . . . A[9] = -3

Индексом может быть не только целое число, но и целое значение переменной или арифметического выражения.
Например: **A[2*i-1] = -2** (при **i=1**).

Перед использованием в программе массив необходимо создать. Обращение к несуществующему элементу вызовет ошибку.

Количество элементов в массиве определяется с помощью функции **len** (length – «длина») . Например: `N = len(A)`

Вывод массива на экран

1 способ. Весь массив выводится как один объект в квадратных скобках, элементы разделяются запятыми.

```
print (A)
```

На экране:

```
[1, 2, 3, 4, 5]
```

2 способ. Вывод элементов с помощью цикла в одной строке через пробел.

```
for i in range(len(A)):
    print (A[i], end=" ")
print()  # переход на новую строку
```

На экране:

```
1 2 3 4 5
```

Вывод массива на экран

3 способ. Вывод элементов с помощью цикла в столбик.

```
for i in range(len(A)):
    print (A[i])
```

На экране:

```
1
2
3
4
5
```

4 способ. Вывод элементов с помощью цикла в столбик с указанием индексов.

```
for i in range(len(A)):
    print ("A[", i, "]=", A[i])
```

На экране:

```
A[ 0 ]= 1
A[ 1 ]= 2
A[ 2 ]= 3
A[ 3 ]= 4
A[ 4 ]= 5
```

Заполнение массивов

1 способ. Создание массива указанием значений элементов

Массив создается перечислением элементов через запятую в квадратных скобках.

```
A = [1, -2, -3, 5, 7]
```

Если все элементы одинаковые, используется следующий оператор.

```
# массив из 5 элементов
# заполненный нулями
A = [0] * 5
```

Заполнение массивов

2 способ. Ввод с клавиатуры (при небольшом количестве элементов)

```
N = 5                              # размер массива в переменной
B = [0] * N                        # заполнение массива нулями
print ("Введите", N, "элементов массива:")
for i in range(N):                 # перебор индексов
    B[i] = int(input())            # ввод числа с клавиатуры
```

Можно в цикл добавить подсказку с индексом вводимого элемента.

```
for i in range(N):                          # перебор индексов
    print ("B[", i, "] = ", end="")         # вывод подсказки
    B[i] = int(input())                     # ввод числа
```

На экране:

```
B[ 0 ] = 1
B[ 1 ] = 2
B[ 2 ] = 3
B[ 3 ] = 4
B[ 4 ] = 5
```

Заполнение массивов

3 способ. Вычисление элементов по формуле (функция от индекса)

```
N = 5                   # размер массива в переменной
C = [0] * N             # заполнение массива нулями
for i in range(N):      # перебор индексов
    C[i] = i**2         # индекс в квадрате
print (C)               # вывод массива
```

На экране:

```
[0, 1, 4, 9, 16]
```

Заполнение массивов

4 способ. Заполнение случайными числами

Функция **`randint(a, b)`** создаёт случайное целое число из отрезка [a, b].

```
N = 5                              # размер массива в переменной
D = [0] * N                        # заполнение массива нулями
from random import randint         # подключение функции randint
for i in range(N):                 # перебор индексов
    D[i] = randint(-5, 5)          # случайные числа от -5 до 5
print (D)                          # вывод массива
```

Возможный результат на экране:

```
[0, -4, -2, 1, 5]
```

Задача 1

Определить средний балл 10 учеников, сдававших ЕГЭ по информатике.

```
# Средний балл учеников
N = 10                  # размер массива
A = [0] * N             # заполнение массива нулями
# Ввод значений элементов массива с клавиатуры
print ("Введите оценки:")
for i in range(N):
    print (i+1, "оценка: ", end="")
    A[i] = int(input())
s = 0                   # нач. знач. суммы
for i in range(N):      # перебор индексов
    s = s + A[i]        # добавление к сумме
sb = s/10               # среднее арифметиче
print ("Средний балл:", sb)
```

```
Введите оценки:
1 оценка: 3
2 оценка: 4
3 оценка: 4
4 оценка: 3
5 оценка: 3
6 оценка: 5
7 оценка: 4
8 оценка: 5
9 оценка: 5
10 оценка: 3
Средний балл: 3.9
```

Задача 2

Подсчитать количество элементов массива, которые больше заданного значения.

```
# Количество элементов массива, соответствующих условию
N = 10
A = [0]*N                       # создание массива
from random import randint      # подключение функции randint
for i in range(N):              # заполнение массива
    A[i] = randint(0, 99)       # случайными числами от 0 до 99
print (A)                       # вывод массива
x = int(input("x = "))          # ввод значения для условия
k = 0                           # начальное значение счетчика
for i in range(N):              # просмотр всех элементов массива
    if A[i] > x:                # если очередной соответ. условию
        k = k+1                 # увеличиваем счетчик
print ("Количество элементов больше данного", k)
```

```
[30, 81, 28, 35, 35, 94, 9, 76, 25, 40]
x = 50
Количество элементов больше данного 3
```

Задача 3

В таблице значений среднесуточной температуры за декаду месяца найти самый холодный день и указать его номер.

```
# Минимальный элемент массива
T = [3,5,6,4,2,9,7,5,4,7]  # создание и вывод массива
print ("Среднесуточная температура за декаду")
print (T)
imin = 0                   # считаем первый элемент минимальн.
for i in range(1, 10):     # просмотр элементов со второго
    if T[i] < T[imin]:     # если очередной меньше минимальн.
        imin = i           # сохраняем его индекс
# вывод максимального элемента и его номера
print ("Минимальная температура: ", T[imin])
print ("День номер: ", imin+1)
```

```
Среднесуточная температура за декаду
[3, 5, 6, 4, 2, 9, 7, 5, 4, 7]
Минимальная температура:  2
День номер:  5
```

Задача 4

Определить, есть ли в данном массиве элемент, значение которого равно заданному числу.

```
# Поиск элемента массива, равного заданному значению
N = 10; A = [0]*N             # создание массива
from random import randint    # подключение функции randint
for i in range(N):            # заполнение массива
    A[i] = randint(0, 99)     # случайными числами от 0 до 99
print (A)                     # вывод массива
x = int(input("x = "))        # ввод значения для поиска
nx = -1                       # несуществующее значение индекса
for i in range(N):            # просмотр всех элементов массива
    if A[i] == x:             # если очередной равен заданному
        nx = i                # сохраняем его индекс
if nx == -1:                  # если индекс не изменился
    print ("Такого значения нет")
else:                         # иначе
    print ("Элемент под номером", nx+1)  # вывод номера элемента
```

```
[24, 26, 14, 17, 8, 7, 12, 39, 50, 64]
x = 50
Элемент под номером 9
```

```
[8, 27, 34, 72, 18, 91, 74, 51, 90, 58]
x = 50
Такого значения нет
```

Сортировка массива

Задача. Упорядочить массив в порядке возрастания значений его элементов.

Алгоритм сортировки методом обмена (метод «пузырька»)

Последовательно сравнить пары соседних чисел и при необходимости поменять их местами, и т.д. для каждой пары. За один проход самое большое число окажется на последнем месте. Затем повторить проход до элемента, уже находящегося на своем месте. И т. д.

	A[1]	A[2]	A[3]	A[4]	A[5]
Исходные значения	5	4	2	3	1
1-й проход					
2-й проход					
3-й проход					
4-й проход					

Сортировка массива

Задача. Упорядочить массив в порядке возрастания значений его элементов.

Алгоритм сортировки методом обмена (метод «пузырька»)

Последовательно сравнить пары соседних чисел и при необходимости поменять их местами, и т.д. для каждой пары. За один проход самое большое число окажется на последнем месте. Затем повторить проход до элемента, уже находящегося на своем месте. И т. д.

	A[1]	A[2]	A[3]	A[4]	A[5]
Исходные значения	5	4	2	3	1
1-й проход	5	4	2	3	1
2-й проход					
3-й проход					
4-й проход					

Сортировка массива

Задача. Упорядочить массив в порядке возрастания значений его элементов.

Алгоритм сортировки методом обмена (метод «пузырька»)

Последовательно сравнить пары соседних чисел и при необходимости поменять их местами, и т.д. для каждой пары. За один проход самое большое число окажется на последнем месте. Затем повторить проход до элемента, уже находящегося на своем месте. И т. д.

	A[1]	A[2]	A[3]	A[4]	A[5]
Исходные значения	5	4	2	3	1
1-й проход	4	2	3	1	5
2-й проход	4	2	3	1	5
3-й проход					
4-й проход					

Сортировка массива

Задача. Упорядочить массив в порядке возрастания значений его элементов.

Алгоритм сортировки методом обмена (метод «пузырька»)

Последовательно сравнить пары соседних чисел и при необходимости поменять их местами, и т.д. для каждой пары. За один проход самое большое число окажется на последнем месте. Затем повторить проход до элемента, уже находящегося на своем месте. И т. д.

	A[1]	A[2]	A[3]	A[4]	A[5]
Исходные значения	5	4	2	3	1
1-й проход	4	2	3	1	5
2-й проход	2	3	1	4	5
3-й проход	2	3	1	4	5
4-й проход					

Сортировка массива

Задача. Упорядочить массив в порядке возрастания значений его элементов.

Алгоритм сортировки методом обмена (метод «пузырька»)

Последовательно сравнить пары соседних чисел и при необходимости поменять их местами, и т.д. для каждой пары. За один проход самое большое число окажется на последнем месте. Затем повторить проход до элемента, уже находящегося на своем месте. И т. д.

	A[1]	A[2]	A[3]	A[4]	A[5]
Исходные значения	5	4	2	3	1
1-й проход	4	2	3	1	5
2-й проход	2	3	1	4	5
3-й проход	2	1	3	4	5
4-й проход	2	1	3	4	5

Сортировка массива

Задача. Упорядочить массив в порядке возрастания значений его элементов.

Алгоритм сортировки методом обмена (метод «пузырька»)

Последовательно сравнить пары соседних чисел и при необходимости поменять их местами, и т.д. для каждой пары. За один проход самое большое число окажется на последнем месте. Затем повторить проход до элемента, уже находящегося на своем месте. И т. д.

	A[1]	A[2]	A[3]	A[4]	A[5]
Исходные значения	5	4	2	3	1
1-й проход	4	2	3	1	5
2-й проход	2	3	1	4	5
3-й проход	2	1	3	4	5
4-й проход	1	2	3	4	5

Сортировка массива

```
# Сортировка элементов массива (метод пузырька по неубыванию)
N = 10; A = [0]*N                # создание массива
from random import randint       # подкл. генератора случайных чисел
for i in range(N):               # заполнение массива
    A[i] = randint(0, 99)        # случайными числами от 0 до 99
print (A)                        # вывод массива
for k in range(1, N-1):          # номер прохода
    for i in range(N-k):         # просмотр за один проход
        if A[i] > A[i+1]:        # если соседние неупорядочены
            A[i], A[i+1] = A[i+1], A[i]    # меняем их местами
    print (A)                    # вывод текущих значений массива
```

```
[92, 47, 84, 49, 24, 73, 98, 19, 65, 90]
[47, 84, 49, 24, 73, 92, 19, 65, 90, 98]
[47, 49, 24, 73, 84, 19, 65, 90, 92, 98]
[47, 24, 49, 73, 19, 65, 84, 90, 92, 98]
[24, 47, 49, 19, 65, 73, 84, 90, 92, 98]
[24, 47, 19, 49, 65, 73, 84, 90, 92, 98]
[24, 19, 47, 49, 65, 73, 84, 90, 92, 98]
[19, 24, 47, 49, 65, 73, 84, 90, 92, 98]
[19, 24, 47, 49, 65, 73, 84, 90, 92, 98]
```

Строковая константа (строка) – произвольная последовательность символов из таблицы Unicode, заключенная в одинарные или двойные кавычки (тип **`str`** – «string»).

Например:

```
'Это строка'
"Это тоже строка"
```

Длина строки – количество символов в строке .

Пустая строка – строка с нулевой длиной (не содержит ни одного символа, обозначается `""`).

Операции со строками

1. Присваивание значения строковой переменной

```
s = "Привет"
```

2. Ввод строки с клавиатуры

```
n = input("Введите имя: ")
```

Примечание: при вводе значения строки кавычки не вводятся.

3. Вывод строки на экран

```
print (n)
```

Примечание: при выводе строки кавычки не выводятся.

4. Объединение строк (конкатенация)

Соединяет несколько строк в одну строку. Обозначается знаком **+**.

Например:

```
"КОМ"+"ПЬЮ"+"ТЕР"   # "КОМПЬЮТЕР"
"10"+"2"            # "102"
```

Операции со строками

5. Определение длины строки

Функция `len(s)` (length – «длина») возвращает длину строки.

Например:

```
len(s)       # 6
len("ЭВМ")   # 3
len("")      # 0
```

6. Выделение отдельного символа

```
s = "Привет"
print (s[2])          # и
```

Примечание: символы нумеруются, начиная с 0.

Операции со строками

7. Выделение части строки (подстроки)

```
s[n:k]
```

Выделяет из строки **s** часть строки от позиции **n** до **k-1**.

```
s[:k]
```

Выделяет из строки **s** часть строки от начала до **k-1**.

```
s[n:]
```

Выделяет из строки **s** часть строки от позиции **n** до конца.

Примечание: символы нумеруются, начиная с 0.

Например:

```
s = "ИНФОРМАТИКА"
print(s[2:7])          # "ФОРМА"
```

Операции со строками

8. Преобразование типов

При необходимости можно преобразовать число в строку или наоборот.

`int(s)` – преобразует строку в целое число.

`float(s)` – преобразует строку в вещественное число.

`str(n)` – преобразует целое или вещественное число в строку.

Примечание: если в строке содержатся символы, не допустимые для чисел, возникнет ошибка.

Например:

```
s1 = "123"                  # строка цифр
i = int(s1)                 # i=123
f = float(s1)               # f=123.0
s2 = str(f)                 # s2="123.0"
print(i, f, len(s2))  # 123 123.0 5
```

Операции со строками

9. Операции с кодами символов

`ord(s)` – возвращает код символа **s**.

`chr(k)` – возвращает символ с кодом **k**.

Примечание: коды из кодовой таблицы Unicode.

Например:

```
print (ord("И"))                          # 1048
print (chr(1048)+chr(1050)+chr(1058))     # ИКТ
```

Операции со строками

10. Сравнение строк

Операции отношения: **==, !=, <, >, <=, >=.**

Сравнение строк производится слева направо до первого несовпадающего символа. Строка считается больше, если первый несовпадающий символ имеет больший код в кодовой таблице (пробел, цифры, латинские заглавные, латинские строчные, русские заглавные, русские строчные). Строки равны, если они совпадают по длине и содержат одни и те же символы.

```
"1STR" < "STR" < "Str" < "str" < "ШАГ" < "Шаг" < "шаг"
```

11. Перебор символов строки

Часто в программах требуется перебирать по одному все символы строки для проведения с ними каких-либо действий. Для этого удобно использовать следующий цикл:

```
for C in S:      # перебор символов в строке S
    <операторы> # очередной символ в перем. C
```

Задача 1

Получить с помощью операций выделения части строки и конкатенации из слова «ИНФОРМАТИКА» слово «ФИРМА».

0 1 2 3 4 5 6 7 8 9 10

ИНФОРМАТИКА

```
a = "ИНФОРМАТИКА"
print(a)
b = a[2] + a[0] + a[4:7]
print(b)
```

```
ИНФОРМАТИКА
ФИРМА
```

Задача 2

В двух строках хранятся фамилия и имя человека. Получить две другие строки в виде «Фамилия_И.», «Имя_Ф.».

```
a = input("Введите фамилию: ")
b = input("Введите имя: ")
print()
c = a + "_" + b[0] + "."
print(c)
d = b + "_" + a[0] + "."
print(d)
```

```
Введите фамилию: Иванов
Введите имя: Петр

Иванов_П.
Петр_И.
```

Задача 3

Из данной строки получить другую строку, состоящую из тех же символов, но в обратном порядке.

```
# Перевертыш (1 способ)
a = input("Введите строку: ")
b = ""                          # нач. знач. результата
for i in range(len(a)):         # перебор символов
    b = a[i]+b                  # присоед. текущий в начало
print("Получен текст:", b)
```

```
# Перевертыш (2 способ)
a = input("Введите текст: ")
b = ""                          # нач. знач. результата
for c in a:                     # перебор символов
    b = c+b                     # присоед. текущий в начало
print("Получен текст:", b)
```

```
Введите текст: АБРАКАДАБРА
Получен текст: АРБАДАКАРБА
```

Задача 4

Подсчитать, сколько раз в данном тексте встречается некоторый заданный символ.

```
# Количество символов в тексте
a = input("Введите текст: ")
b = input("Какой символ подсчитать? ")
k = 0                    # начальное знач. счетчика
for c in a:              # перебор символов
    if c==b:             # если заданный символ
        k = k+1          # увеличиваем счетчик на 1
print("В тексте таких символов", k)
```

```
Введите текст: АБРАКАДАБРА
Какой символ подсчитать? А
В тексте таких символов 5
```

Задача 5

В данном тексте после каждого символа вставить некоторый заданный символ.

```
# Вставка символов после каждого в тексте
a = input("Введите текст: ")
b = input("Какой символ вставлять? ")
c = ""              # начальное знач. результата
for d in a:         # перебор символов
    c = c+d+b       # присоединяем текущий и заданный
print("Получен текст: ")
print(c)
```

```
Введите текст: АБРАКАДАБРА
Какой символ вставлять? *
Получен текст:
А*Б*Р*А*К*А*Д*А*Б*Р*А*
```

Задача 6

В данном тексте удалить некоторый заданный символ.

```
# Удаление заданного символа
a = input("Введите текст: ")
b = input("Какой символ удалять? ")
c = ""                  # начальное знач. результата
for d in a:             # перебор символов
    if d!=b:            # если текущий не равен заданному
        c = c+d         # присоединяем текущий символ
print("Получен текст: ")
print(c)
```

```
Введите текст: АБРАКАДАБРА
Какой символ удалять? А
Получен текст:
БРКДБР
```