Программирование (Python)

§ 21. Массивы

Что такое массив?

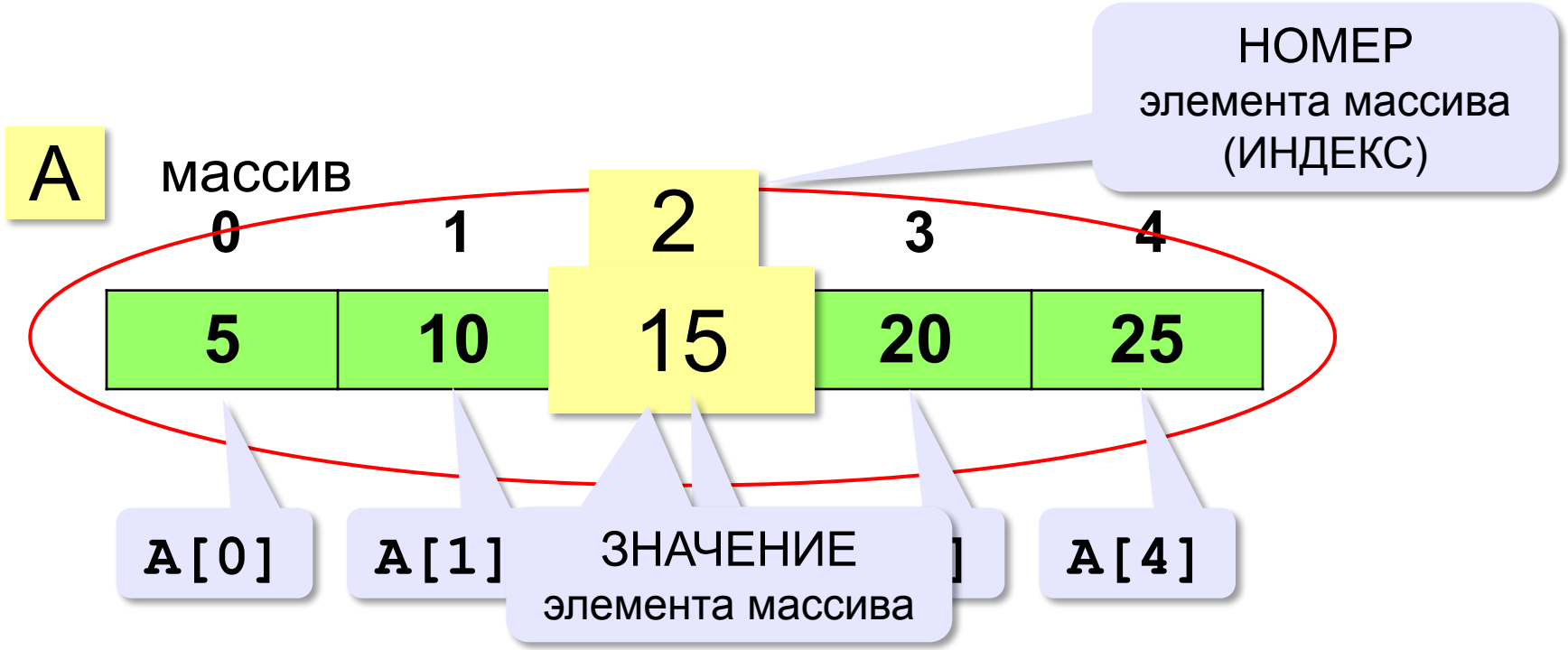
Как ввести 10000 переменных?

Массив – это группа переменных одного типа, расположенных в памяти рядом (в соседних ячейках) и имеющих общее имя.

Надо:

- выделять память
- записывать данные в нужную ячейку
- читать данные из ячейки

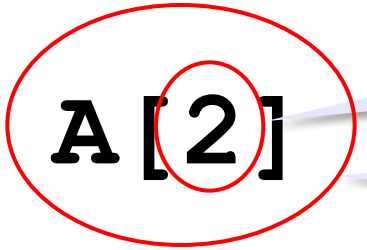
Обращение к элементу массива



Индекс элемента — это значение, которое указывает на конкретный элемент массива.

! Нумерация с нуля!

Обращение к элементу массива



ИНДЕКС элемента массива: 2

ЗНАЧЕНИЕ элемента массива

0	1	2	3	4
23	12	7	43	51

```
i = 1
A[2] = A[i] + 2*A[i-1] + A[2*i+1]
print( A[2]+A[4] )
```

Что получится?

```
A[2] = A[1] + 2*A[0] + A[3]
print( A[2]+A[4] )
```

```
101
152
```

Создание массива

```
A = [11, 22, 35, 41, 53]
```

11	22	35	41	53

```
A = [11, 22] + [35, 41] + [53]
```

```
A = [11]*5
```

11	11	11	11	11

Что неверно?

```
A = [1, 2, 3, 4, 5]
x = 1
print( A[x-3] )
A[x+4] = A[x-1] + A[2*x]
```

Что плохо?

```
print( A[-2] )
A[5] = A[0] + A[2]
```

Выход за границы массива — это обращение к элементу с индексом, который не существует в массиве.

Перебор элементов массива

```
N = 10
A = [0]*N   # память уже выделена
```

Перебор элементов: просматриваем все элементы массива и, если нужно, выполняем с каждым из них некоторую операцию.

```
for i in range(N):
    # здесь работаем с A[i]
```

Заполнение массива

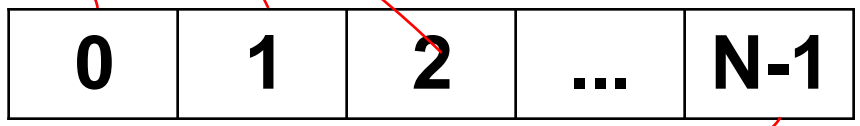
```
[0, 2, 3, …, N-1]
```

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

? Что произойдёт?

В развёрнутом виде

```
A[0] = 0
A[1] = 1
A[2] = 2
...
A[N-1] = N-1
```



В стиле Python:

```
A = [ i for i in range(N) ]
```

Заполнение массива в обратном порядке

N	...	3	2	1

```
A[0] = N
A[1] = N-1
A[2] = N-2
...
A[N-1] = 1
```

```
X = N
for i in range(N):
    A[i] = X
    X = X - 1
```

? Как меняется X?

X = N, N-1, …, 2, 1

начальное значение

уменьшение на 1

Заполнение массива в обратном порядке

N	...	3	2	1

`A[i] = X`

? Как связаны `i` и `X`?

i	X
0	N
1	N-1
2	N-2
...	...
N-1	1

+1

–1

```
for i in range(N):
  A[i] = N - i
```

В стиле Python:

```
A = [ N-i
   for i in range(N) ]
```

! Сумма `i` и `X` не меняется!

```
i + X = N
X = N - i
```

Вывод массива на экран

Весь массив сразу:

```
print( A )
```

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

По одному элементу:

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

в столбик

или так:

```
for x in A:
  print( x )
```

для всех элементов в массиве A

Как вывести в строчку?

```
for x in A:
  print( x, end=" " )
```

пробел между элементами

Вывод массива на экран (Python)

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

```
print( *A )
```

⟷

```
print(1, 2, 3, 4, 5)
```

разбить список на элементы

```
1 2 3 4 5
```

Ввод с клавиатуры

```
for i in range(N):
  A[i] = int(input())
```

Что плохо?

или так:

```
A=[int(input())
        for i in range(N)]
```

С подсказкой для ввода:

```
for i in range(N):
  s = "A[" + str(i) + "]="
  A[i] = int(input(s))
```

```
A[0]=5
A[1]=12
A[2]=34
A[3]=56
A[4]=13
```

Ввод с клавиатуры (Python)

Ввод всех чисел в одной строке:

```
data = input()     # "1 2 3 4 5"
s = data.split() #["1","2","3","4","5"]
A = [ int(x) for x in s ]
                   # [1,2,3,4,5]
```

или так:

```
A = [int(x) for x in input().split()]
```

Заполнение случайными числами

из библиотеки (модуля) random

взять функцию **randint**

```
from random import randint
N = 10        # размер массива
A = [0]*N     # выделить память
for i in range(N):
  A[i] = randint(20,100)
```

В краткой форме:

```
from random import randint
N = 10
A = [ randint(20,100)
      for i in range(N) ]
```

В других языках программирования

Паскаль:

```
const N = 10;
var A: array[0..N-1] of integer;
...
for i:=0 to N-1 do
  A[i] = i;
for i:=0 to N-1 do
  write(A[i], ' ');
```

объявление массива

В других языках программирования

C++:

```
int A[N], i;
for (i = 0; i < N; i++)
  A[i] = i;
for (i = 0; i < N; i++)
  cout << A[i] << " ";
```

Нумерация элементов всегда с нуля!

Программирование (Python)

Алгоритмы обработки массивов

Сумма элементов массива

Задача. Найти сумму элементов массива из N элементов.

Какие переменные нужны?

5	2	8	3	1

```
Sum = 0
for i in range(N):
    Sum += A[i]
print( Sum )
```

i	Sum
	0
0	5
1	7
2	15
3	18
4	19

В стиле Python:

```
print( sum(A) )
```

Сумма элементов массива (Python)

Задача. Найти сумму элементов массива A.

```
Sum = 0
for x in A:
    Sum += x
print( Sum )
```

для всех элементов из A

! Не нужно знать размер!

или так:

```
print( sum(A) )
```

встроенная функция

Сумма не всех элементов массива

Задача. Найти сумму чётных элементов массива.

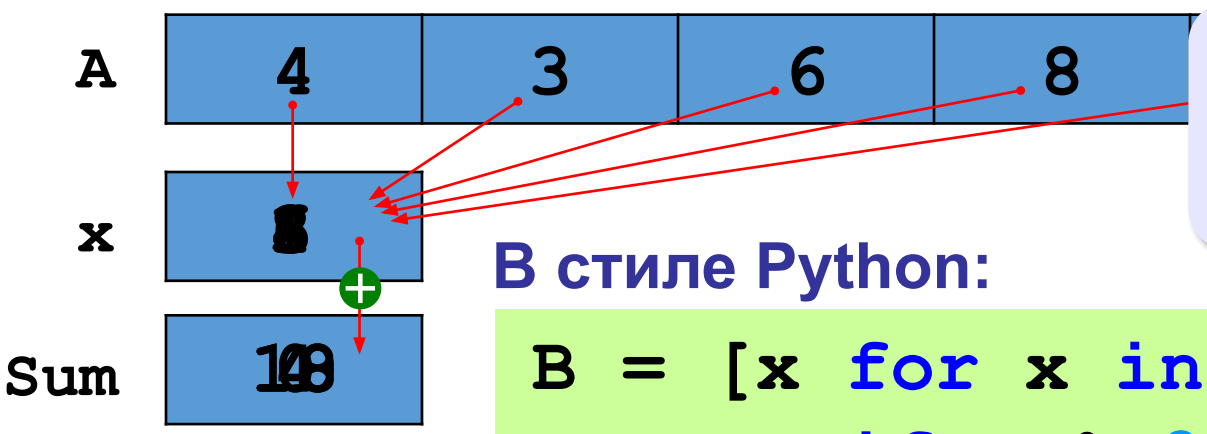
Что делаем с нечётными?

```
Sum = 0
for i in range(N):
    if A[i] % 2 == 0:
        Sum += A[i]
print( Sum )
```

Сумма не всех элементов массива

Задача. Найти сумму чётных элементов массива.

```
Sum = 0
for x in A:
   if x % 2 == 0:
      Sum += x
print( Sum )
```



отбираем в новый массив все нужные значения

В стиле Python:

```
B = [x for x in A
       if x % 2 == 0]
print ( sum(B) )
```

Подсчёт элементов по условию

Задача. Найти количество чётных элементов массива.

Какие переменные нужны?

```
count = 0
for i in range(N):
    if A[i] % 2 == 0:
        count += 1
print( count )
```

переменная-счётчик

Что тут делаем?

Подсчёт элементов по условию (Python)

Задача. Найти количество чётных элементов массива.

```
count = 0
for x in A:
  if x % 2 == 0:
    count += 1
print( count )
```

В стиле Python:

```
B = [x for x in A
       if x % 2 == 0]
print ( len(B) )
```

размер массива

Среднее арифметическое

Задача. Найти среднее арифметическое элементов массива, которые больше 180 (рост в см).

```
Sum = 0
for x in A:
   if x > 180:
      Sum += x
print( Sum/N )
```

Что плохо?

Среднее арифметическое

Задача. Найти среднее арифметическое элементов массива, которые больше 180 (рост в см).

Какие переменные нужны?

```
Sum = 0
count = 0
for x in A:
   if x > 180:
      count += 1
      Sum += x
print( Sum/count )
```

Что тут делаем?

Среднее арифметическое (Python)

Задача. Найти среднее арифметическое элементов массива, которые больше 180 (рост в см).

```
B = [ x for x in A
            if x > 180]
print ( sum(B)/len(B) )
```

отбираем нужные

Обработка потока данных

Задача. С клавиатуры вводятся числа, ввод завершается числом 0. Определить, сколько было введено положительных чисел.

1) нужен счётчик
2) счётчик увеличивается
3) нужен цикл
4) это цикл с условием (число шагов неизвестно)

? Когда увеличивать счётчик?

? Какой цикл?

```
счётчик = 0
пока не введён 0:
   если введено число > 0 то
      счётчик:= счётчик + 1
```

Обработка потока данных

```
count = 0
x = int(input())
while x != 0:
   if x > 0:
      count += 1
   x = int(input())
print( count )
```

откуда взять x?

? Что плохо?

Найди ошибку!

```
count  =0
x = int(input())
while x != 0:
   if x > 0:
      count += 1
prx = int(input())
```

Найди ошибку!

```
count = 0          ut())
while x == 0:
   if x >!=:
      count += 1
   x = int(input())
print( count )
```

Обработка потока данных

Задача. С клавиатуры вводятся числа, ввод завершается числом 0. Найти сумму введённых чисел, оканчивающихся на цифру "5".

1) нужна переменная для суммы
2) число добавляется к сумме, если оно заканчивается на "5"
3) нужен цикл с условием

Как это записать?

```
сумма = 0
пока не введён 0:
 если число оканчивается на "5" то
    сумма:= сумма + число
```

```
if x % 10 == 5:
```

Обработка потока данных

Задача. С клавиатуры вводятся числа, ввод завершается числом 0. Найти сумму введённых чисел, оканчивающихся на цифру "5".

```
sum = 0
x = int(input())
while x != 0:
    if x % 10 == 5:
        sum += x
    x = int(input())
print( sum )
```

Чего не хватает?

Найди ошибку!

```
sum = 0
x = int(input())
   if x % 10 == 5:
      sum += x
   x = int(input())
print( sum )
```

Перестановка элементов массива

Как поменять местами значения двух переменных *a* и *b*?

вспомогательная переменная

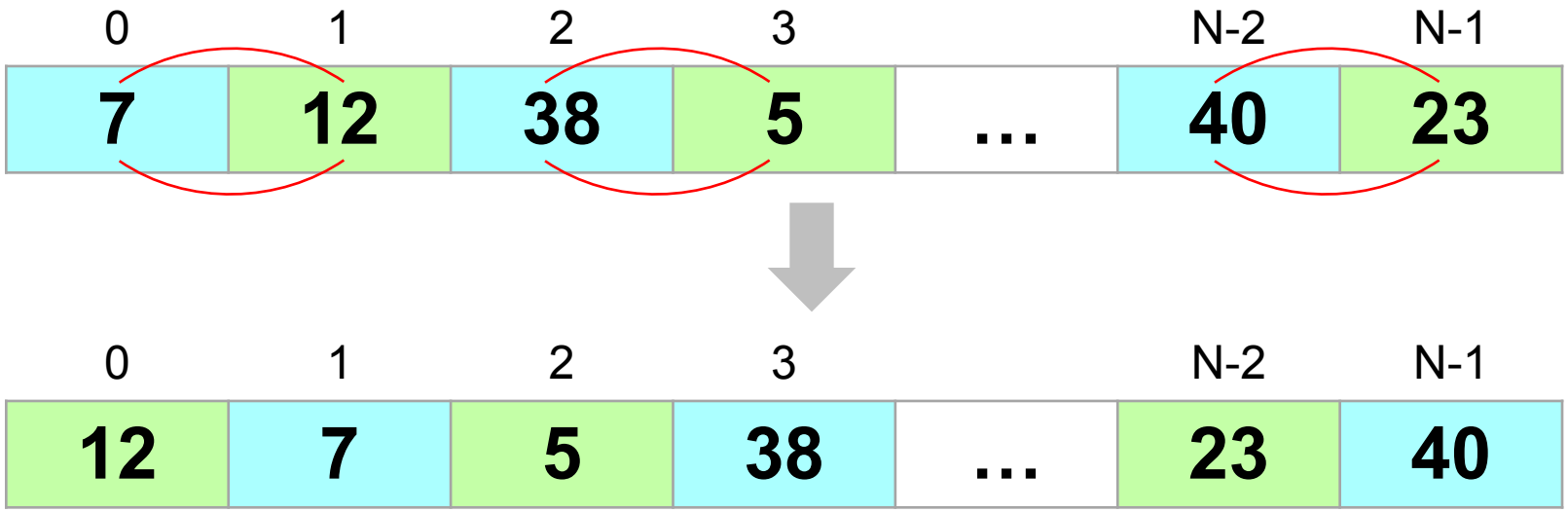
```
c = a
a = b
b = c
```

элементы массива:

```
c = A[i]
A[i] = A[k]
A[k] = c
```

Перестановка пар соседних элементов

Задача. Массив A содержит чётное количество элементов N. Нужно поменять местами пары соседних элементов: 0-й с 1-м, 2-й — с 3-м и т. д.



Перестановка пар соседних элементов

```
for i in range(N):
   поменять местами A[i] и A[i+1]
```

Что плохо?

0	1	2	3	4	5
7	12	38	5	40	23
12	7	38	5	40	23
12	38	7	5	40	23
12	38	5	7	40	23
12	38	5	40	7	23
12	38	5	40	23	7

выход за границы массива

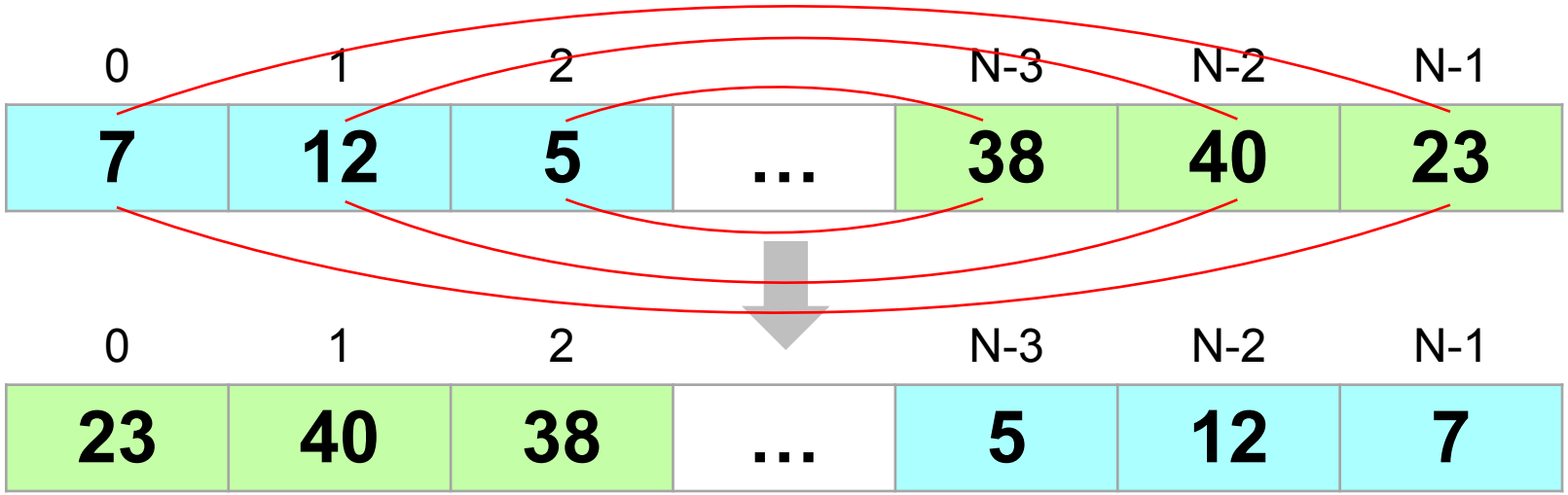
Перестановка пар соседних элементов

```
for i in range(0,N,2):
  # переставляем A[i] и A[i+1]
  c = A[i]
  A[i] = A[i+1]
  A[i+1] = c
```

A[1]↔A[2], A[3]↔A[4], …, A[N-1]↔A[N]

Реверс массива

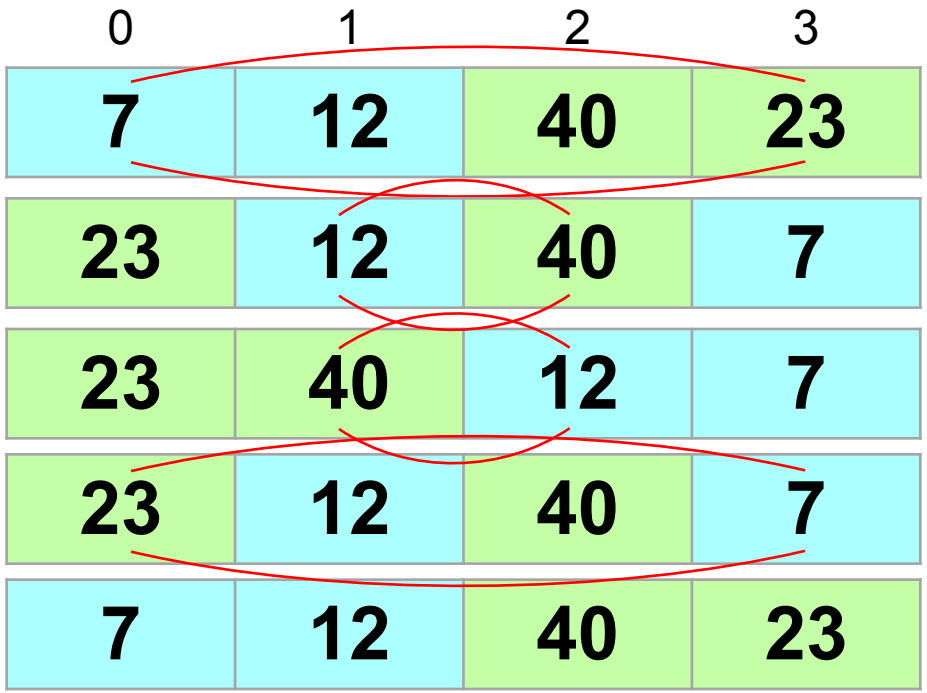
Задача. Переставить элементы массива в обратном порядке (выполнить *реверс*).




```
A[0]↔A[N-1]        0+N-1 = N-1
A[1]↔A[N-2]        1+N-2 = N-1
A[i]↔A[N-1-i]      i+??? = N-1
A[N-1]↔A[0]        N-1+0 = N-1
```

Реверс массива

```
for i in range (N % 2):
    поменять местами A[i] и A[N+1-i]
```



Что плохо?

Как исправить?

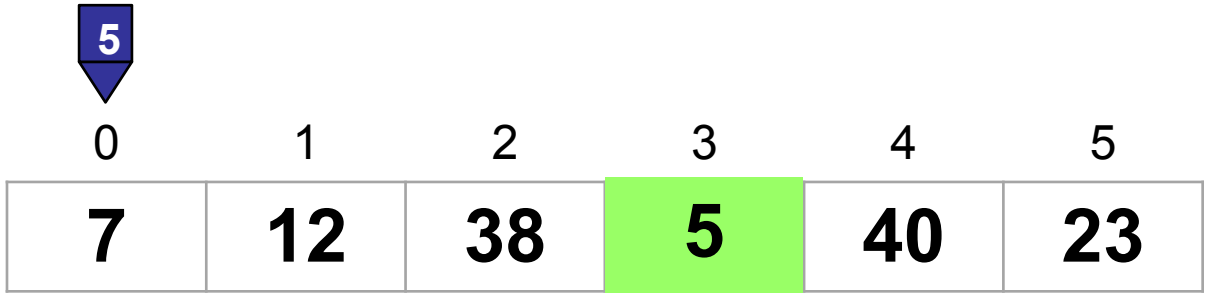
Программирование (Python)

Поиск и сортировка

Линейный поиск в массиве

Задача. Найти в массиве элемент, равный X, и его номер.

X = 5




```
i = 0
while A[i]!=X:
   i += 1
print("A[", i, "]=", X)
```

? Что плохо?

? Если искать 4?

! Нельзя выходить за границы массива!

Линейный поиск в массиве

не выходим за границу

```
i = 0
while i<=N and A[i]!=X:
   i += 1
if i <= N:
   print( "A[",i,"]=",X )
else:
   print( "Не нашли!" )
```

? Как проверить, нашли или нет?

Досрочный выход из цикла

Задача. Найти в массиве элемент, равный X, и его **номер**.

```
nX = 0 # номер элемента
for i in range(N):
  if A[i]==X:
    nX = i # запомнить номер
    break
if nX > 0:
  print( "A[", nX, "]=", X )
else:
  print( "Не нашли!" )
```

нашли!

сразу выйти из цикла

Поиск в массиве

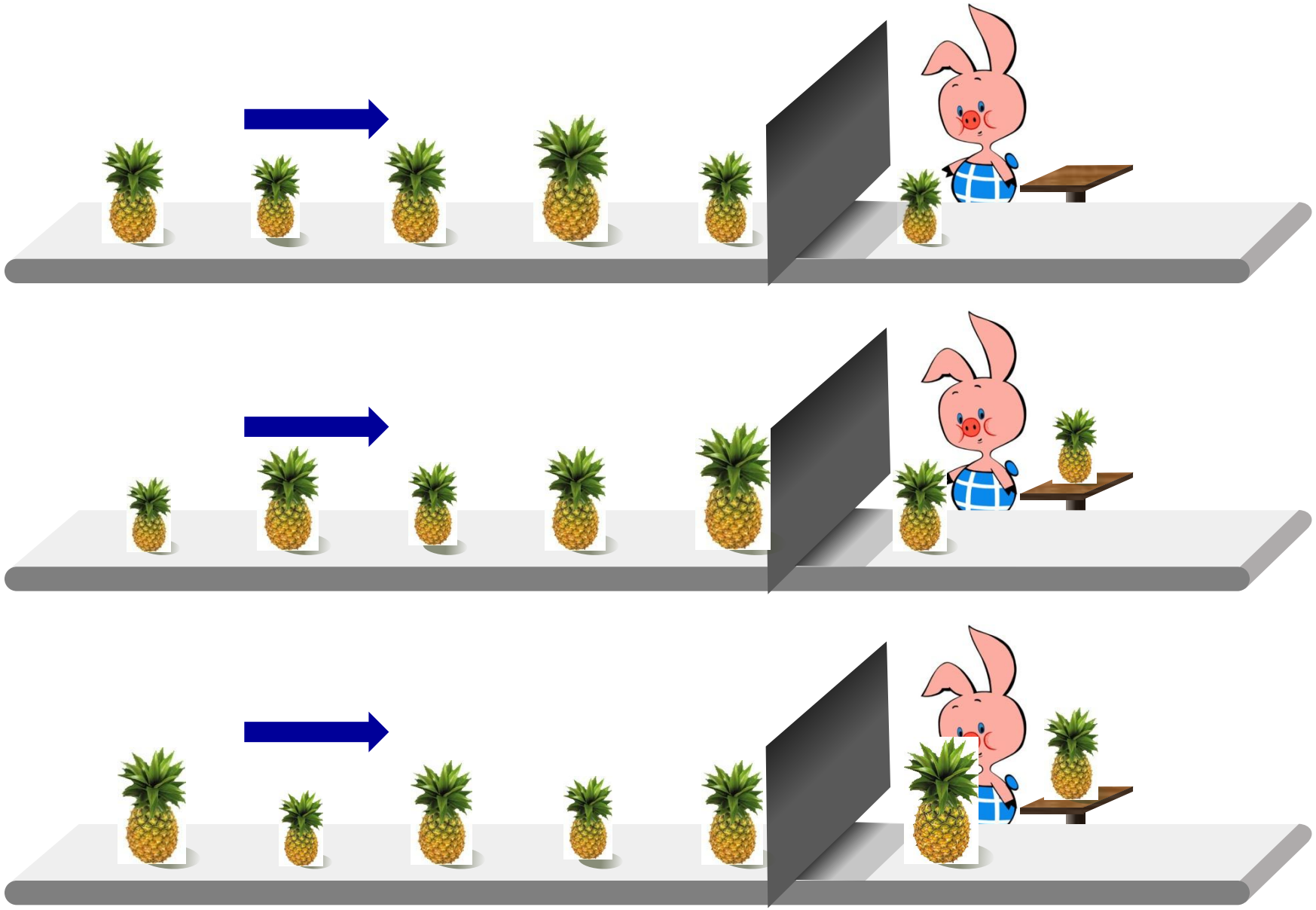
Варианты в стиле Python:

```
for i in range ( N ):
  if A[i] == X:
    print ( "A[", i, "]=", X, sep = "" )
    break
else:
  print ( "Не нашли!" )
```

если не было досрочного выхода из цикла

```
if X in A:
  nX = A.index(X)
  print ( "A[", nX, "]=", X, sep = "" )
else:
  print ( "Не нашли!" )
```

Поиск максимального элемента



Поиск максимального элемента

? Какие переменные нужны?

```
for i in range(N):
   if A[i] > M:
      M = A[i]
print( M )
```

? Чего не хватает?

? Какое начальное значение взять для **M**?

1) **M** – значение, которое заведомо меньше всех элементов массива

   или

2) **M = A[0]** (или любой другой элемент)

максимальный не меньше, чем **A[0]**

Поиск максимального элемента

```
M = A[0]
for i in range(1,N):
   if A[i] > M:
      M = A[i]
print( M )
```

начинаем с `A[1]`, так как `A[0]` мы уже посмотрели

Как найти минимальный?

Поиск максимального элемента (Python)

```
M = A[0]
for x in A:
    if x > M:
        M = x
print( M )
```

перебрать все элементы в массиве A

Не нужно знать размер!

```
print( max(A) )
```

```
print( min(A) )
```

Номер максимального элемента

Задача. Найти в массиве максимальный элемент и его номер.

Какие переменные нужны?

```
M = A[0]; nMax = 0
for i in range(1,N):
   if A[i] > M:
      M = A[i]
      nMax = i
print( "A[", nMax, "]=", M )
```

Можно ли убрать одну переменную?

Номер максимального элемента

Если знаем **nMax**, то **M=A[nMax]**!

```
M = A[0]; nMax = 0
for i in range(1,N):
   if A[i]>A[nMax]:
      M = A[i]
      nMax = i
print( "A[", nMax, "]=", A[nMax] )
```

Максимальный элемент и его номер

Вариант в стиле Python:

```
M = max(A)
nMax = A.index(M)
print ( "A[", nMax, "]=", M )
```

номер заданного элемента (первого из…)

Максимальный не из всех

Задача. Найти в массиве максимальный из отрицательных элементов.

```
M = A[0]
for i in range(1,N):
   if A[i]<0 and A[i]>M:
      M = A[i]
print( M )
```

Что плохо?

Как исправить?

0	1	2	3	4
5	–2	8	3	–1

M = 5

Максимальный не из всех

Задача. Найти в массиве максимальный из отрицательных элементов.

```
M = A[0]
for i in range(1,N):
   if A[i]<0:
      if M>=0 or A[i]>M:
         M = A[i]
print( M )
```

сначала записали неотрицательный!

? Если нет отрицательных?

Максимальный не из всех (Python)

Задача. Найти в массиве максимальный из отрицательных элементов.

```
B = [ x for x in A
          if x < 0]
print( max(B) )
if len(B):
  print( max(B) )
else:
  print("Нет таких!")
```

отбираем нужные

?

Если нет отрицательных?

```
if len(B)!= 0:
```

Сортировка выбором

? Где должен стоять минимальный элемент?

- нашли минимальный, поставили его на первое место

```
c = A[nMin]
A[0],A[nMin] = A[nMin],A[0]
A[0] = c
```

? Как?

- из оставшихся нашли минимальный, поставили его на второе место и т.д.

? Что дальше?

5	–2	8	3	–1
–2	5	8	3	–1
–2	–1	8	3	5

Сортировка выбором

```
for i in range(N-1):
    # ищем минимальный среди A[i]..A[N-1]

  nMin = i
  for j in range(i+1,N):
    if A[j] < A[nMin]:
      nMin = j
  # переставляем A[i] и A[nMin]
  A[i],A[nMin] = A[nMin],A[i]
```

не трогаем те, которые уже поставлены

? Почему цикл `N-1` раз?

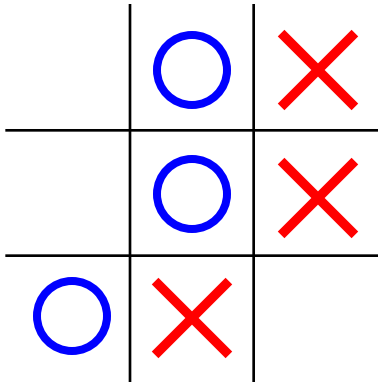
Решение в стиле Python:

```
A.sort()
```

Программирование (Python)

Матрицы (двумерные массивы)

Что такое матрица?



нет знака

нолик

крестик

строка 2,
столбец 3

? Как закодировать?

Матрица — это прямоугольная таблица, составленная из элементов одного типа (чисел, строк и т.д.).

Каждый элемент матрицы имеет два индекса – номера строки и столбца.

Создание матриц

Матрица – это массив масивов!

```
A = [[-1, 0, 1],
     [-1, 0, 1],
     [0, 1, -1]]
```

перенос на другую строку внутри скобок

или так:

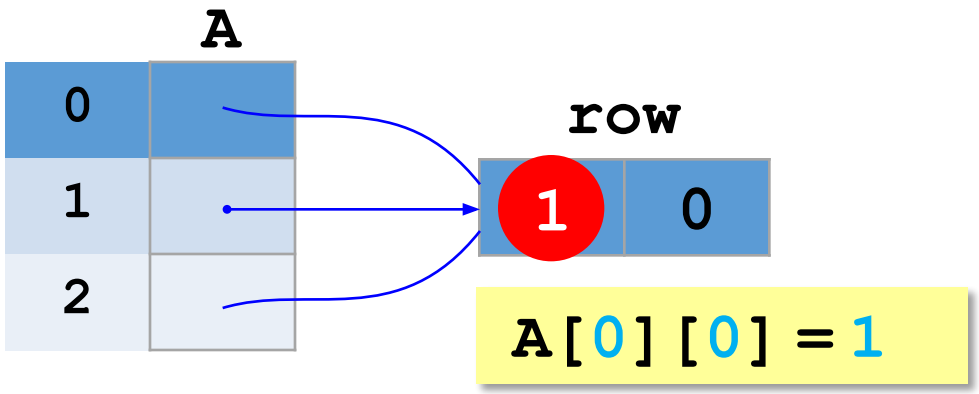
```
A = [[-1, 0, 1], [-1, 0, 1], [0, 1, -1]]
```

Нумерация элементов с нуля!

Создание матриц

Нулевая матрица:

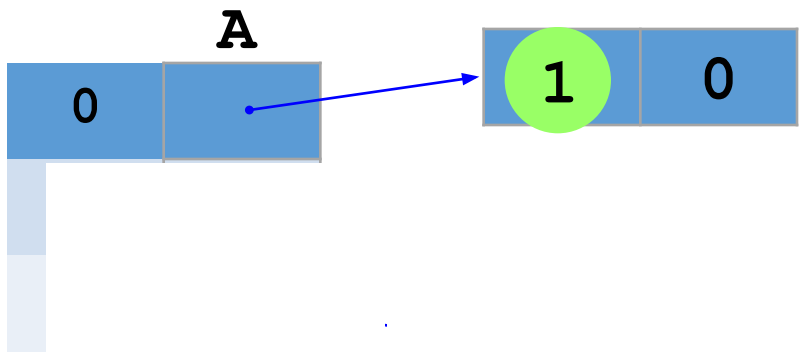
```
N = 3
M = 2
row = [0]*M
A = [row]*N
```



A[0][0] = 1

а правильно так:

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



A[0][0] = 1

Вывод матриц

```
print ( A )
```

```
[[1, 2, 3], [4, 5, 6], [7, 8, 9]]
```

```
def printMatrix( A ):
  for row in A:
    for x in row:
      print ( "{:4d}".format(x), end="" )
    print ()
```

```
   1   2   3
   4   5   6
   7   8   9
```

Зачем форматный вывод?

Простые алгоритмы

Заполнение случайными числами:

```
from random import randint
for i in range(N):
  for j in range(M):
    A[i][j] = randint ( 20, 80 )
    print ( "{:4d}".format(A[i][j]),
            end = "" )
  print()
```

Вложенный цикл!

Суммирование:

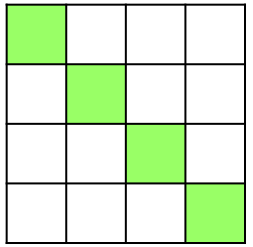
```
s = 0
for i in range(N):
  for j in range(M):
    s += A[i][j]
print ( s )
```

```
s = 0
for row in A:
  s += sum(row)
print ( s )
```

Перебор элементов матрицы

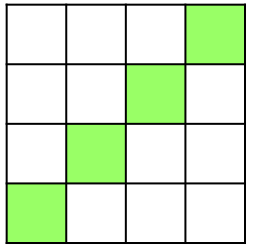
Главная диагональ:

```
for i in range(N):
  # работаем с A[i][i]
```



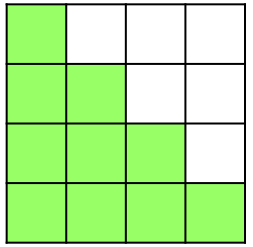
Побочная диагональ:

```
for i in range(N):
  # работаем с A[i][N-1-i]
```



Главная диагональ и под ней:

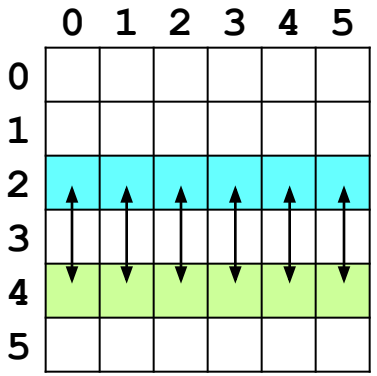
```
for i in range(N):
  for j in range(i+1):
    # работаем с A[i][j]
```



Перестановка строк

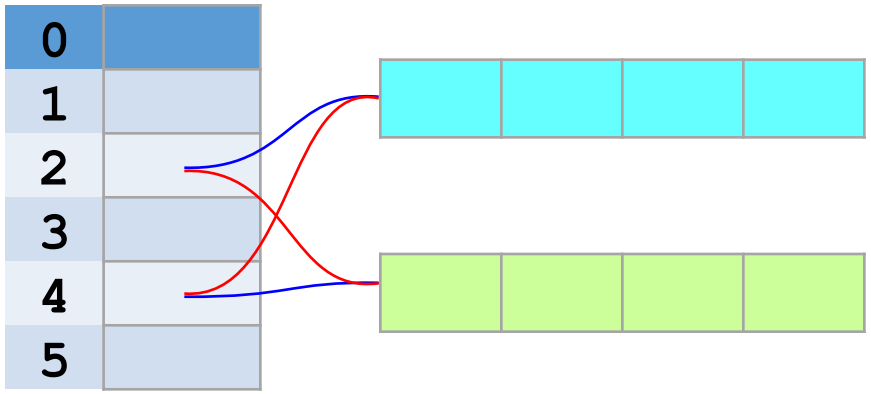
2-я и 4-я строки:

```
for j in range(M):
    A[2,j],A[4,j] = A[4,j],A[2,j]
    A[2,j] = A[4,j]
    A[4,j] = c
```



Решение в стиле Python:

```
A[2],A[4] = A[4],A[2]
```



Перестановка столбцов

2-й и 4-й столбцы:

```
for i in range(N):
    A[i,2],A[i,4] = A[i,4],A[i,2]
    A[i,2] = A[i,4]
    A[i,4] = c
```