Программирование на языке Паскаль Часть II

Тема 1. Массивы

Массивы

Массив – это группа однотипных элементов, имеющих общее имя и расположенных в памяти рядом.

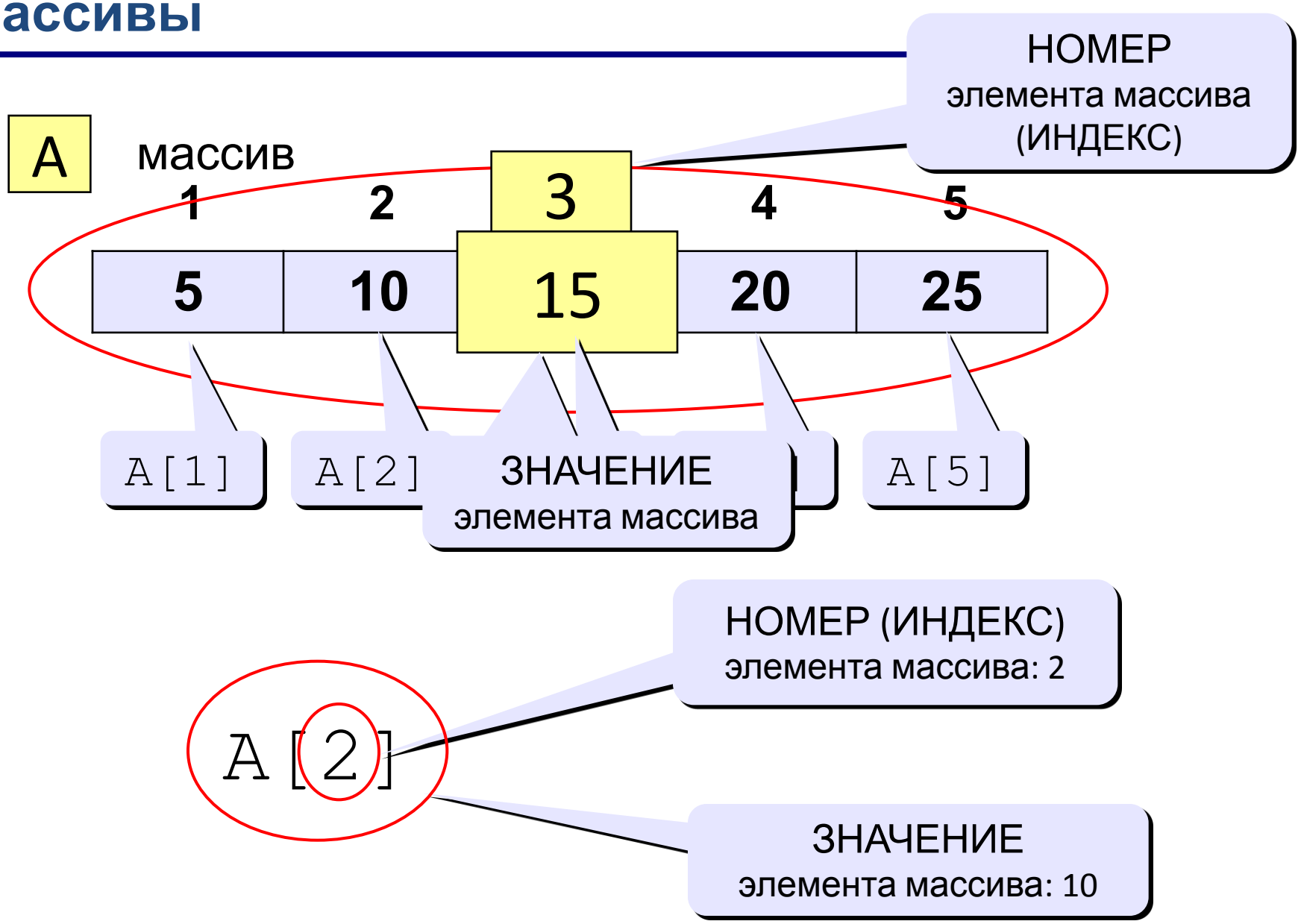
Особенности:

- все элементы имеют **один тип**
- весь массив имеет **одно имя**
- все элементы расположены в памяти **рядом**

Примеры:

- список учеников в классе
- квартиры в доме
- школы в городе
- данные о температуре воздуха за год

Массивы

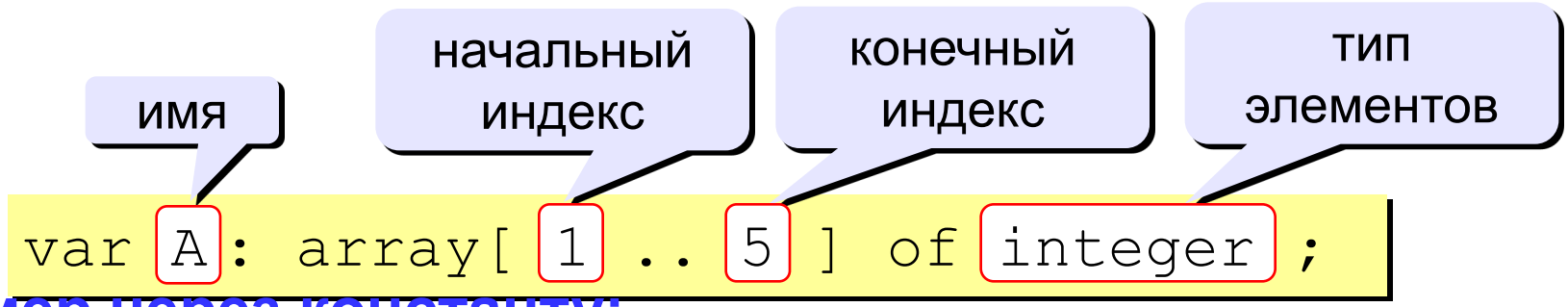

НОМЕР
элемента массива
(ИНДЕКС)
A
массив
1
2
3
4
5
5
10
15
20
25
A[1]
A[2]
ЗНАЧЕНИЕ
элемента массива
A[5]
НОМЕР (ИНДЕКС)
элемента массива: 2
A[2]
ЗНАЧЕНИЕ
элемента массива: 10

Объявление массивов

Зачем объявлять?

- определить **имя** массива
- определить **тип** массива
- определить **число элементов**
- выделить **место в памяти**

Массив целых чисел:



Размер через константу:

```
const N=5;
var A: array[1..N] of integer;
```

Объявление массивов

Массивы других типов:

```
var X, Y: array [1..10] of real;
    C: array [1..20] of char;
```

Другой диапазон индексов:

```
var Q: array [0..9] of real;
    C: array [-5..13] of char;
```

Ин...

```
var A: array ['A'..'Z'] of real;
    B: array [False..True] of integer;
...
   A['C'] := 3.14259*A['B'];
   B[False] := B[False] + 1;
```

Что неправильно?

```
var a: array [1..10] of integer;
...
  A[5] := 4.5;
```

```
var a: array ['a'..'z'] of integer;
...
  A['b'] := 15;
```

```
var a: array [0..9] of integer;
...
  A[10] := 'x';
```

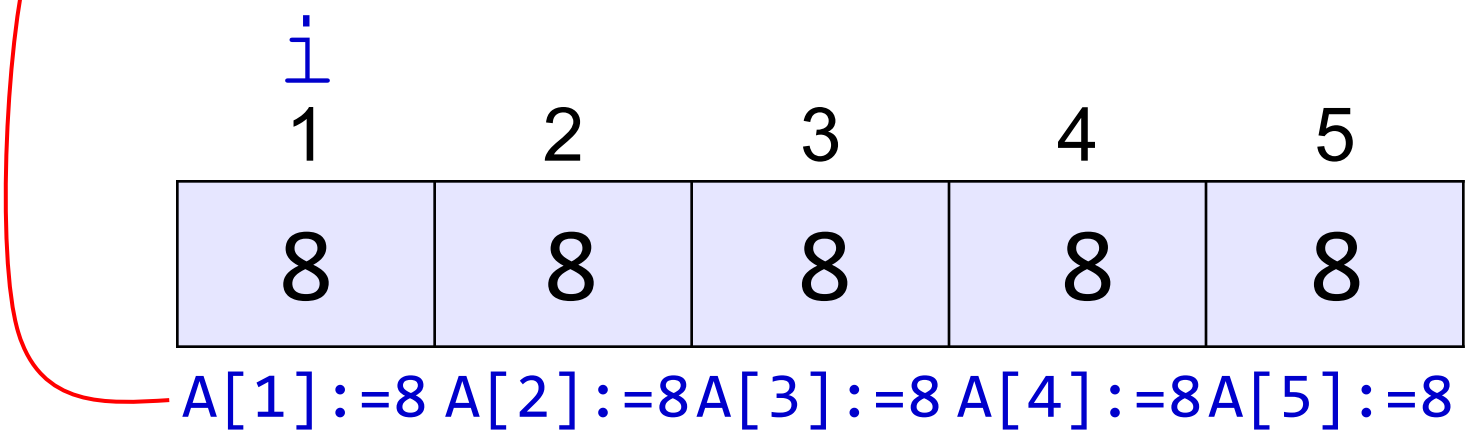
Заполнение массива

Объявление:

```
const N = 5;
var A: array[1..N] of integer;
    i: integer;
```

Заполнение одинаковыми числами:

```
for i:=1 to N do begin
  A[i]:= 8;
end;
```



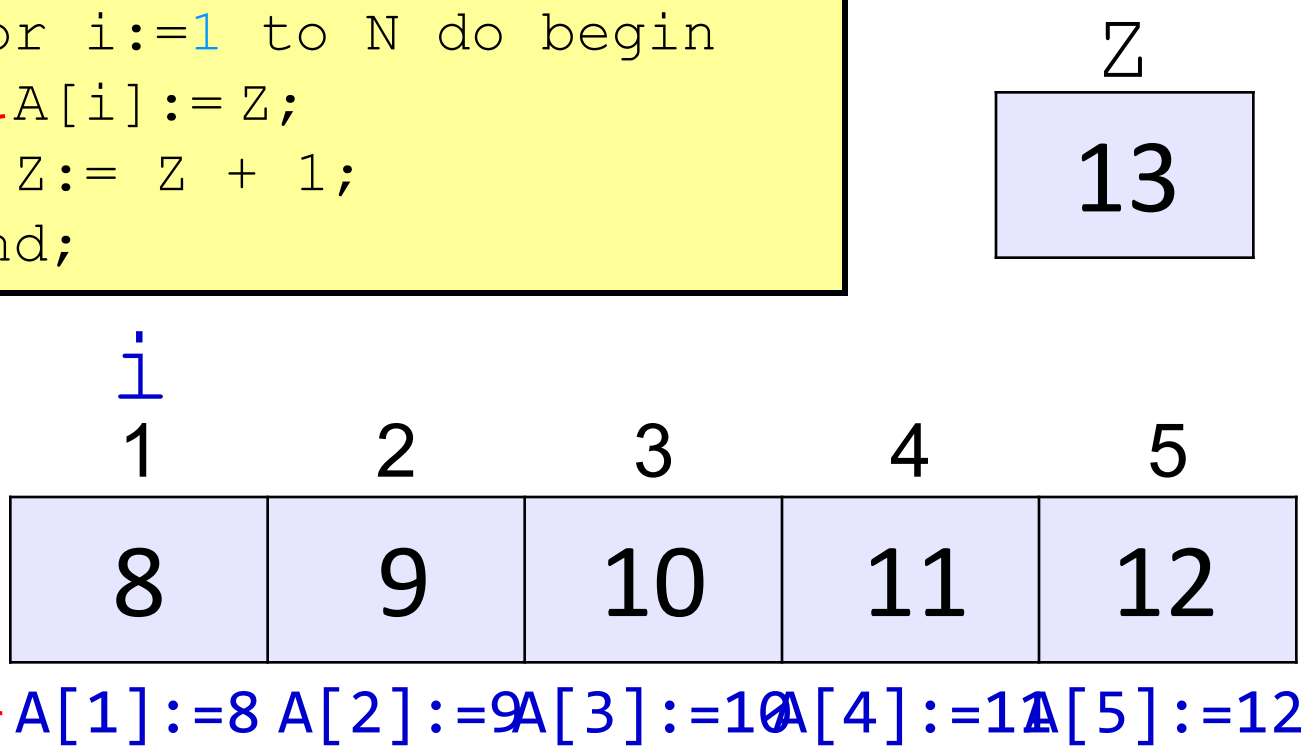
Заполнение массива

Объявление:

```
const N = 5;
var A: array[1..N] of integer;
    i: integer;
```

Заполнение последовательными числами:

```
Z:= 8;
for i:=1 to N do begin
  A[i]:= Z;
  Z:= Z + 1;
end;
```



Заполнение массива

Заполнение последовательными числами:

```
Z:= 8;
for i:=1 to N do begin
  A[i]:=Z;
  Z:= Z + 1;
end;
```

$$Z = i + 7$$

```
for i:=1 to N do begin
  A[i]:=i + 7;
```

i	Z

? Как связаны **i** и **Z**?

Массивы

Объявление:

```
const N = 5;
var a: array[1..N] of integer;
    i: integer;
```

Ввод с клавиатуры:

```
for i:=1 to N do begin
  write('a[', i, ']=');
  read ( a[i] );
end;
```

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

? Почему write?

По

```
for i:=1 to N do a[i]:=a[i]+1;
```

Вы

```
writeln('Массив A:');
for i:=1 to N do
  write(a[i]:4);
```

```
Массив A:
   6   13   35   57   14
```

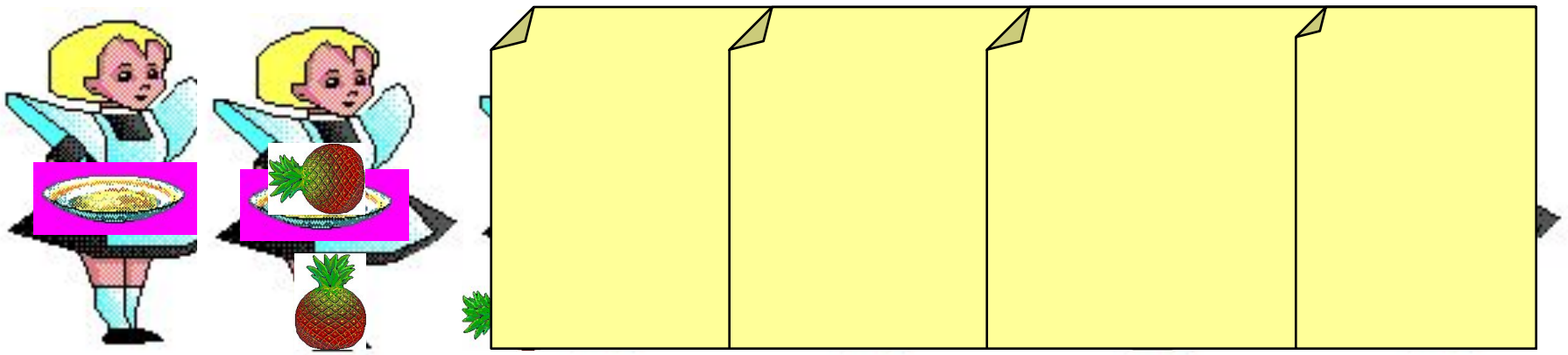
Программирование на языке Паскаль
Часть II

Тема 2. Максимальный элемент массива

Максимальный элемент

Задача: найти в массиве максимальный элемент.

Алгоритм:



Псевдокод:

```
{ считаем, что первый элемент – максимальный }
for i:=2 to N do
  if a[i] > { максимального } then
    { запомнить новый максимальный элемент a[i] }
```

? Почему цикл от i=2?

Максимальный элемент

Дополнение: как найти номер максимального элемента?

```
                { считаем, что первый – максимальный }
iMax := 1;
for i:=2 to N do      { проверяем все остальные }
  if a[i] > a[iMax] then { нашли новый максимальный }
  begin
                      { запомнить a[i] }
    iMax := i;        { запомнить i }
  end;
```

? Как упростить?

По номеру элемента `iMax` всегда можно найти его значение `a[iMax]`. Поэтому везде меняем max на `a[iMax]` и убираем переменную `max`.

Программа

```
program qq;
const N = 5;
var a: array [1..N] of integer;
    i, iMax: integer;
begin
  { здесь нужно ввести массив с клавиатуры }
  iMax := 1; {считаем, что первый – максимальный}
  for i:=2 to N do      { проверяем все остальные}
    if a[i] > a[iMax] then { новый максимальный}
      iMax := i;                { запомнить i }
  writeln; {перейти на новую строку}
  writeln('Максимальный элемент a[',
          iMax, ']=', a[iMax]);
end.
```

Программирование на языке Паскаль
Часть II

Тема 3. Обработка массивов

Случайные процессы

Случайно…

1)встретить друга на улице
2)разбить тарелку
3)найти 10 рублей
4)выиграть в лотерею

Случайный выбор:

1)жеребьевка на соревнованиях
2)выигравшие номера в лотерее

Как получить случайность?

Случайные числа на компьютере

Электронный генератор

- нужно специальное устройство
- нельзя воспроизвести результаты

Псевдослучайные числа – обладают свойствами случайных чисел, но каждое следующее число вычисляется по заданной формуле.

Метод середины квадрата (Дж. фон Нейман)

564321

в квадрате

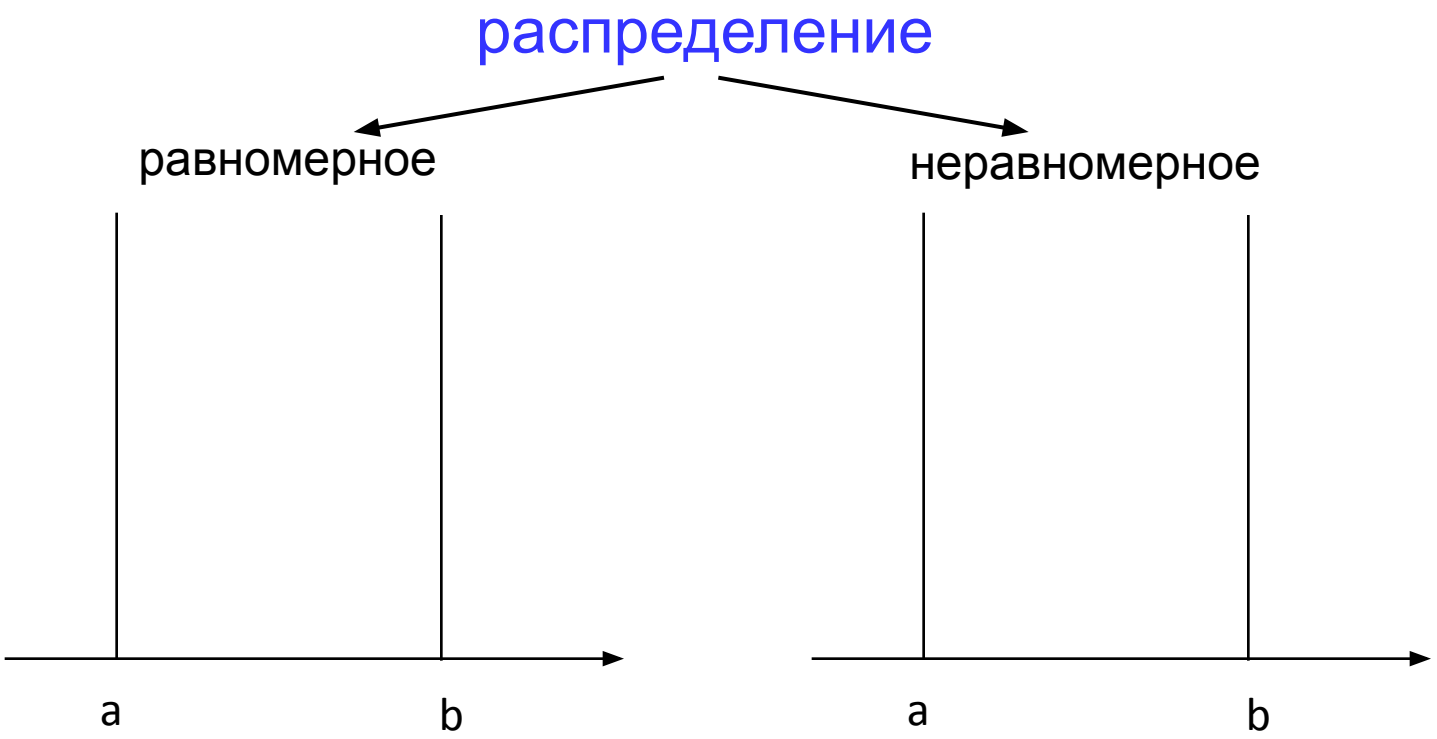
318458191041

209938992481

малый период (последовательность повторяется через 10^6 чисел)

Распределение случайных чисел

Модель: снежинки падают на отрезок [a,b]

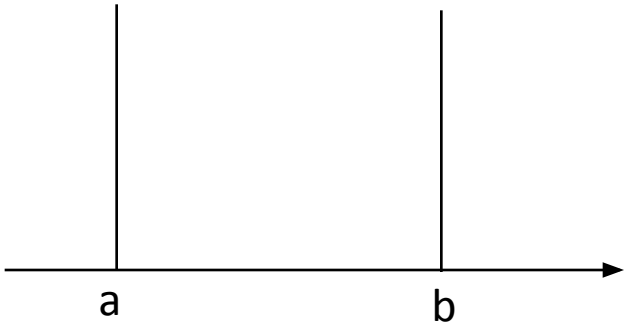


? Сколько может быть разных распределений?

Распределение случайных чисел

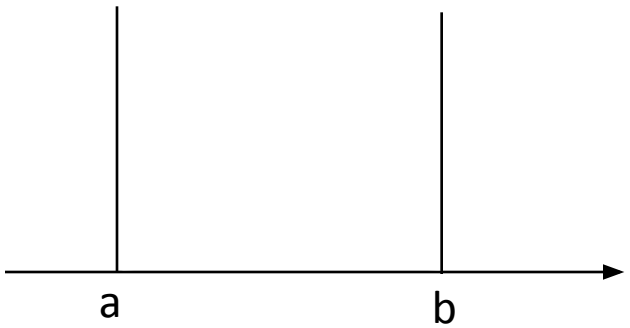
Особенности:

- распределение – это характеристика **всей последовательности**, а не одного числа
- **равномерное** распределение одно, компьютерные датчики случайных чисел дают равномерное распределение
- неравномерных – много
- любое неравномерное можно получить с помощью равномерного



$$x = \frac{x_1 + x_2}{2}$$

равномерное распределение



$$x = \frac{x_1 + x_2 + \boxtimes + x_{12}}{12}$$

неравномерное распределение

Генератор случайных чисел в Паскале

Целые числа в интервале [0,N):

```
var x: integer;
...
x := random ( 100 );   { интервал [0,99] }
```

Вещественные числа в интервале [0,1)

```
var x: real;
...
x := random;           { интервал [0,1) }
```

Генератор случайных чисел в Паскале

Целые числа на отрезке [a,b]:

```
var x: integer;
...
x := random ( N );
```

[0,N-1]

```
x := a + random ( N );
```

[a,a+N-1]

? Какой отрезок?

? Как выбрать N?

```
b = a + N - 1
N = b - a + 1
```

```
x := a + random ( b-a+1 );
```

Заполнение массива случайными числами

```
const N = 5;
var A: array [1..N] of integer;
    i: integer;
begin
  writeln('Исходный массив:');
  for i:=1 to N do begin
    A[i] := random(100) + 50;
    write(A[i]:4);
  end;
  ...
```

случайные числа в интервале [50,150)

? Зачем сразу выводить?

Подсчет элементов

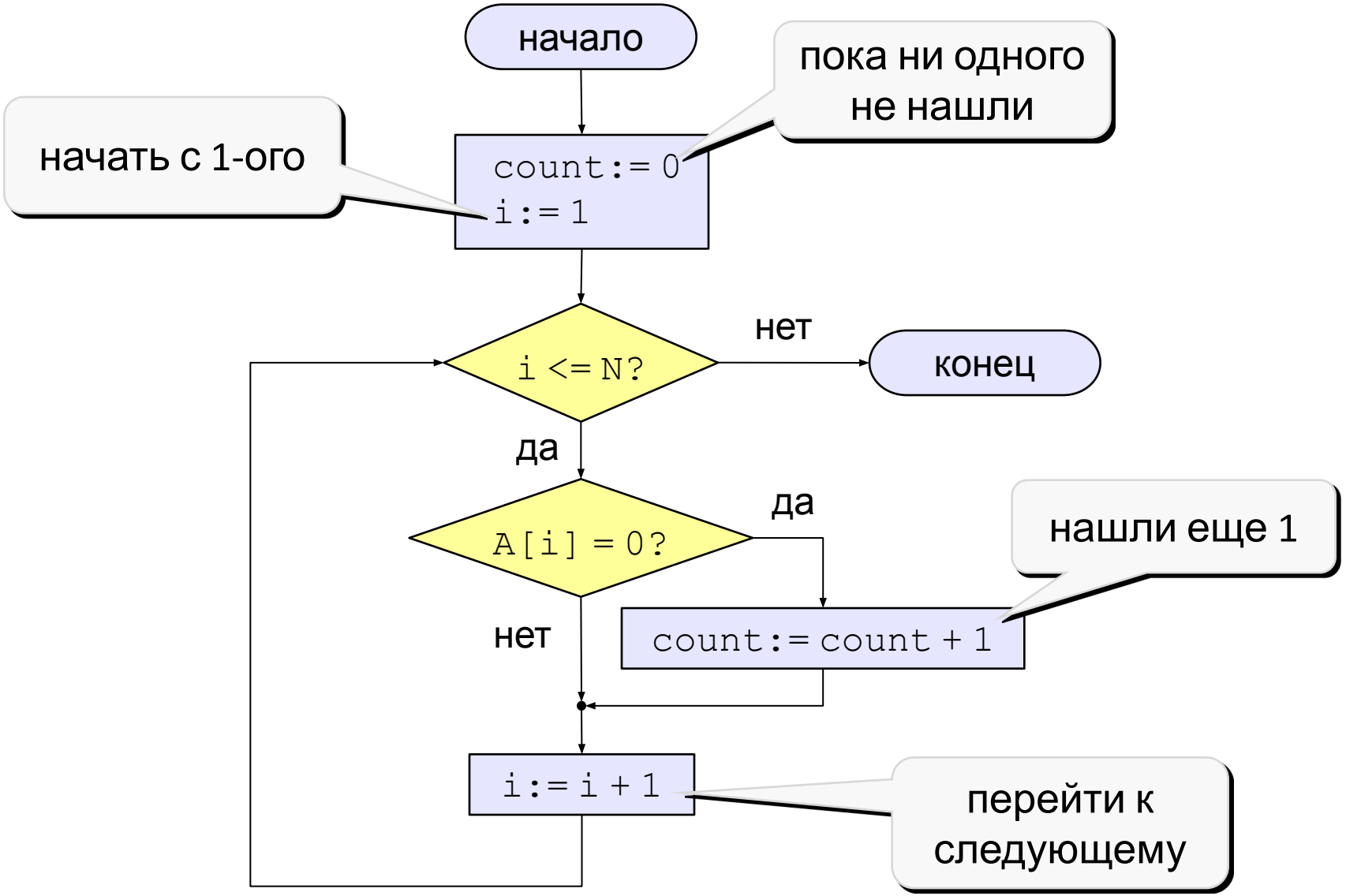
Задача: заполнить массив случайными числами в интервале `[-1,1]` и подсчитать количество нулевых элементов.

Идея: используем переменную-счётчик.

Решение:

1)записать в счётчик ноль
2)просмотреть все элементы массива:
   если очередной элемент = 0,
     то увеличить счётчик на 1
3)вывести значение счётчика

Подсчет элементов



Подсчет элементов

```
program qq;
const N = 5;
var A: array [1..N] of integer;
    i, count: integer;
begin
  { здесь надо заполнить массив }
  count:= 0;
  for i:=1 to N do
    if A[i] = 0 then count:= count + 1;
  writeln('Нулевых элементов: ', count);
end.
```

перебираем все элементы массива

Сумма выбранных элементов

Задача: заполнить массив случайными числами в интервале `[-10,10]` и подсчитать сумму положительных элементов.

Идея: используем переменную **S** для накопления суммы.

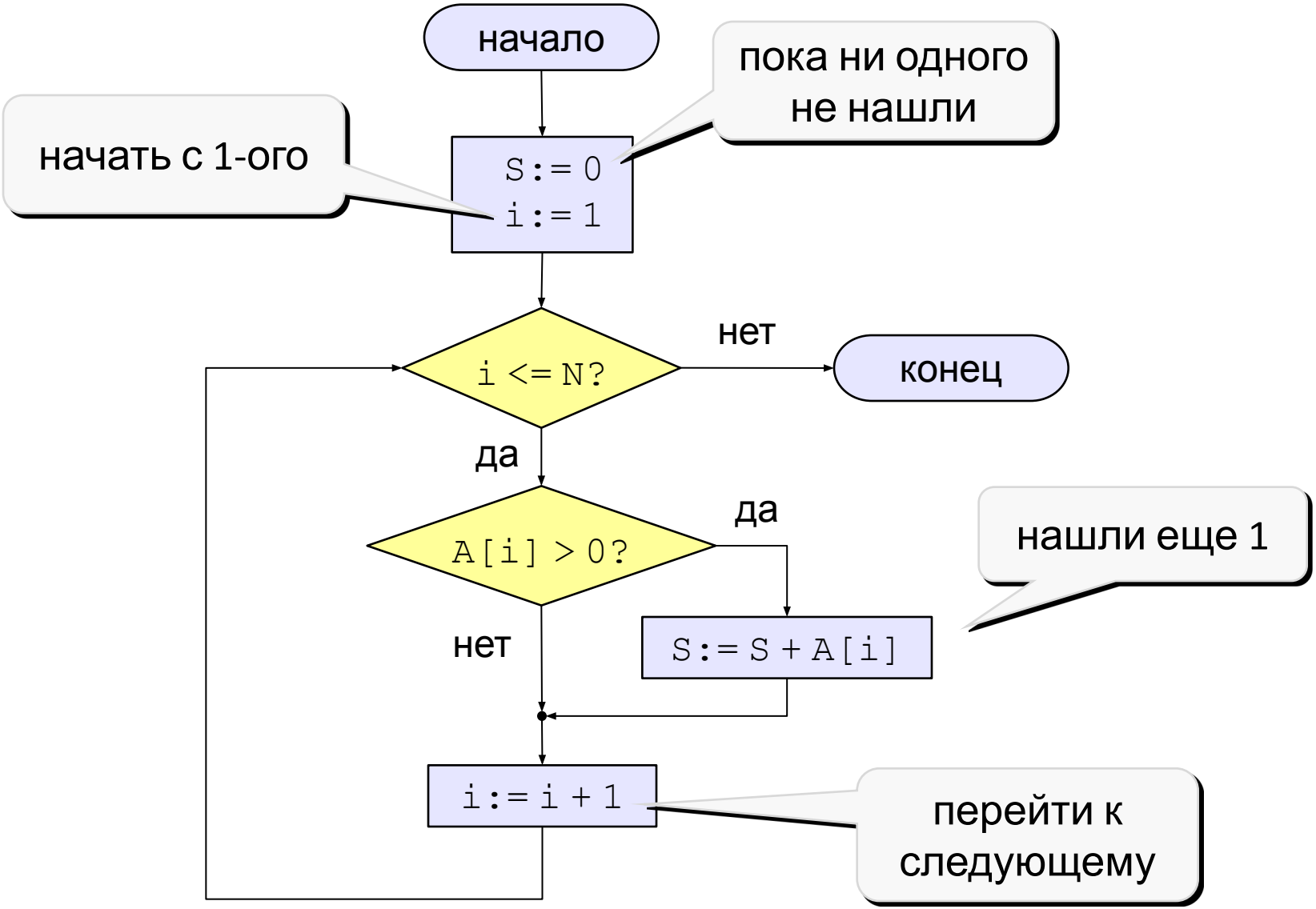
```
S:=0    S:=A[1]    S:=A[1]+A[2]
S:=A[1]+A[2]+A[3]  ➡  S:=A[1]+A[2]+…+A[N]
```

Решение:

1)записать в переменную **S** ноль
2)просмотреть все элементы массива:
если очередной элемент > 0,
то добавить к сумме этот элемент
3)вывести значение суммы

```
S:=S+A[i]
```

Сумма выбранных элементов



Сумма выбранных элементов

```
program qq;
const N = 5;
var A: array [1..N] of integer;
    i, S: integer;
begin
  { здесь надо заполнить массив }
  S:=0;
  for i:=1 to N do
    if A[i] > 0 then S:= S + A[i];
  writeln('Сумма полож. элементов: ', S);
end.
```

перебираем все элементы массива

Поиск в массиве

Задача – найти в массиве элемент, равный **X**, или установить, что его нет.

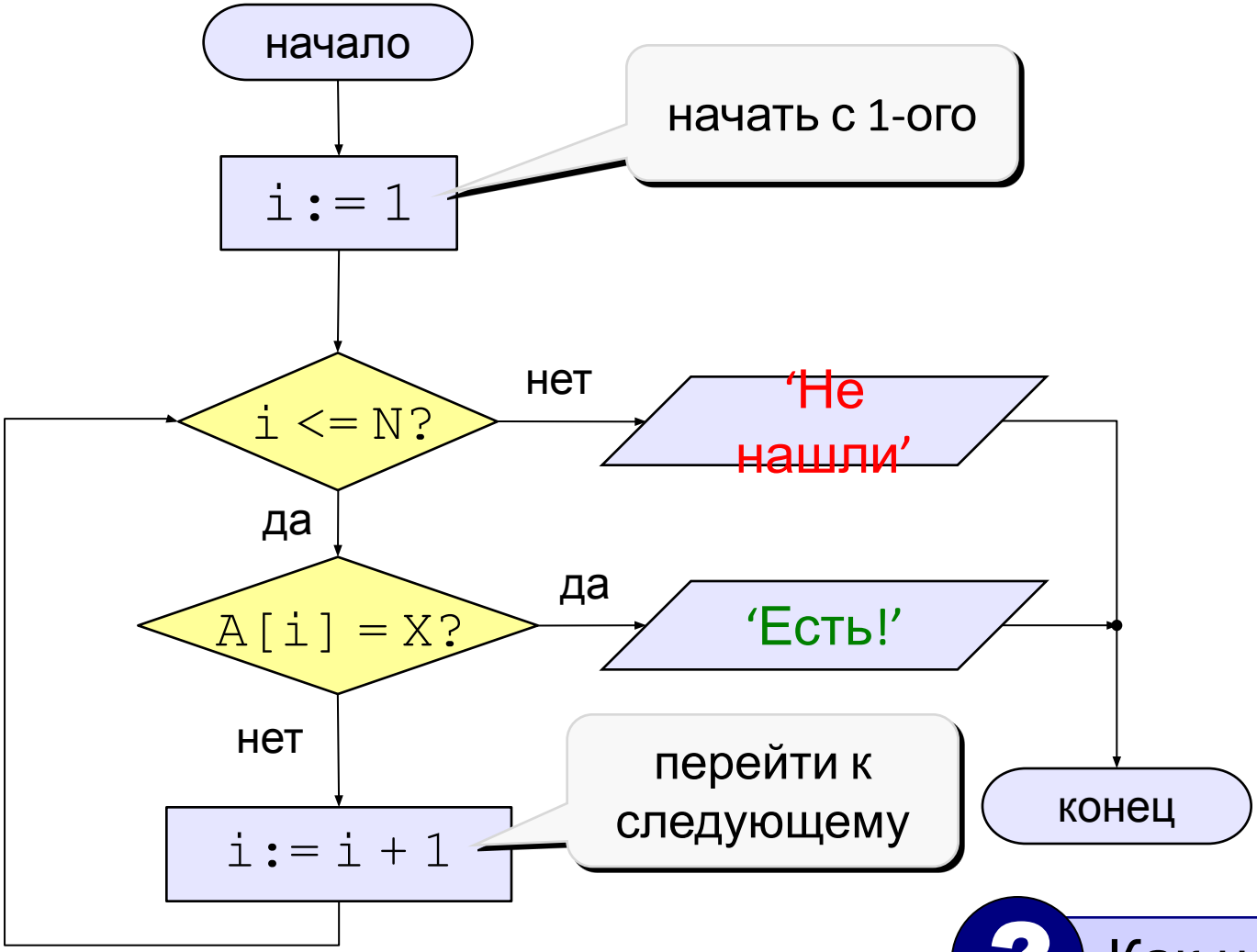
Пример: если в классе ученик с фамилией Пупкин?

Алгоритм:

1)начать с 1-ого элемента (`i:=1`)

2)если очередной элемент (`A[i]`) равен X, то
закончить поиск
иначе перейти к следующему элементу:

Поиск элемента, равного X



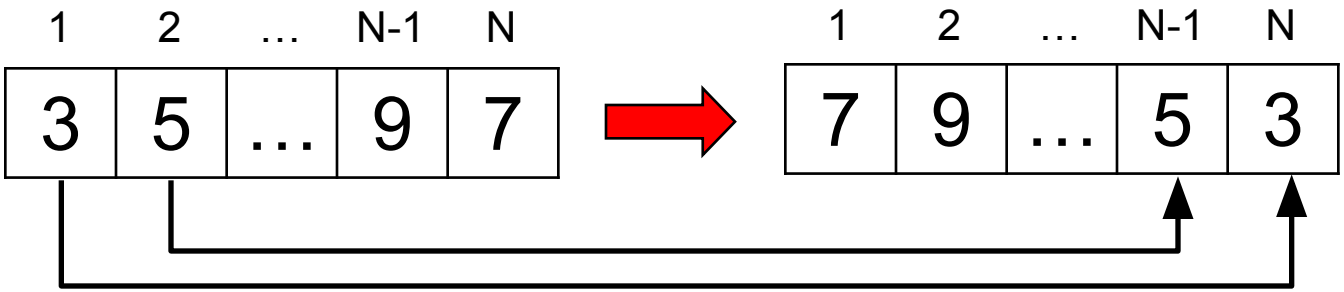
? Как найти номер?

Поиск элемента в массиве

```
program qq;
const N=5;
var a:array[1..N] of integer;
    i, X: integer;
begin
  { здесь надо заполнить массив }
  i:=1;
  while (i<=N) and (A[i]<>X) do
    i:=i+1;
  if i <= N then
       writeln('A[', i, ']=', X)
  else writeln('Не нашли...');
end.
```

Реверс массива

Задача: переставить элементы массива в обратном порядке.



Алгоритм:

сумма индексов N+1

поменять местами **A[1]** и **A[N]**, **A[2]** и **A[N-1]**, …

Псевдокод:

```
for i:=1 to N div 2 do
  { поменять местами A[i] и A[N+1-i] }
```

Что неверно?

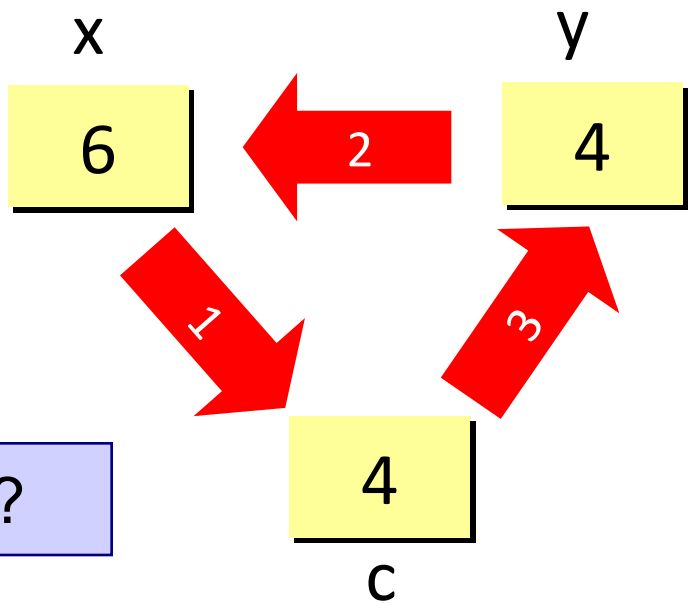
Как переставить элементы?

Задача: поменять местами содержимое двух чашек.

Задача: поменять местами содержимое двух ячеек памяти.

```
x := y;
y := x;
```

```
c := x;
x := y;
y := c;
```



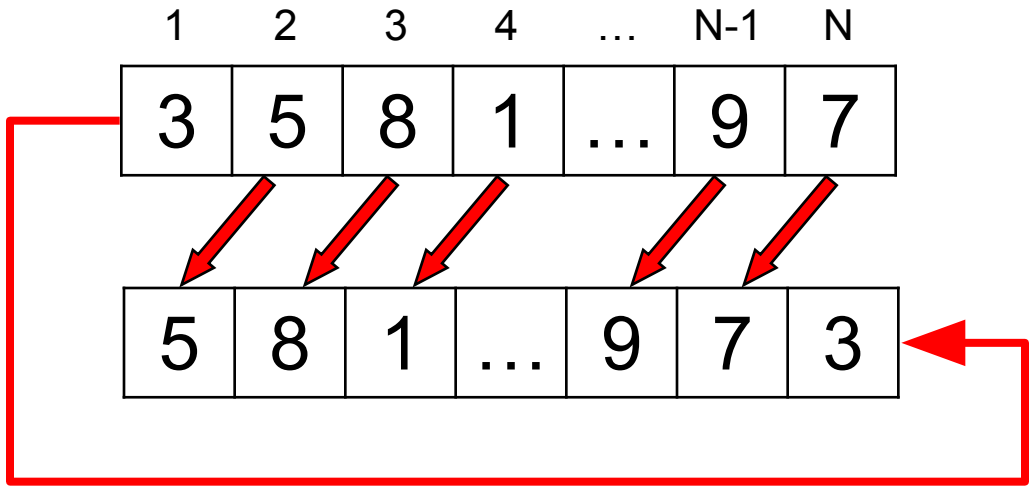
? Можно ли обойтись без **c**?

Программа

```
program qq;
const N = 10;
var A: array[1..N] of integer;
    i, c: integer;
begin
  { заполнить массив }
  { вывести исходный массив }
  for i:=1 to N div 2 do begin
     c:=A[i]; A[i]:=A[N+1-i]; A[N+1-i]:=c;
   end;
  { вывести полученный массив }
end.
```

Циклический сдвиг

Задача: сдвинуть элементы массива влево на 1 ячейку, первый элемент становится на место последнего.



Алгоритм:

```
A[1]:=A[2]; A[2]:=A[3];… A[N-1]:=A[N];
```

Цикл:

почему не N?

```
for i:=1 to N-1 do
  A[i]:=A[i+1];
```

? Что неверно?

Программа

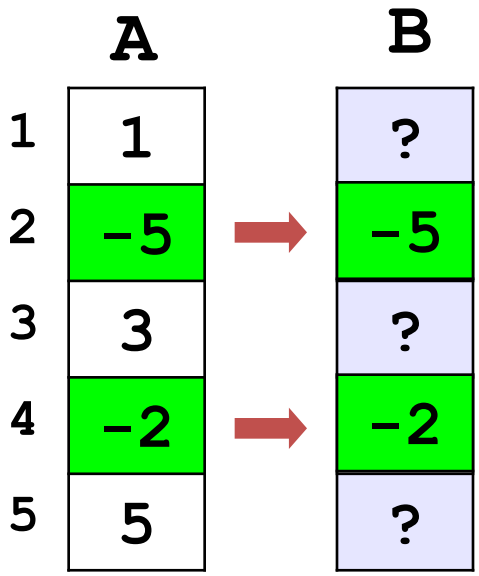
```
program qq;
const N = 10;
var A: array[1..N] of integer;
    i, c: integer;
begin
  { заполнить массив }
  { вывести исходный массив }
  c := A[1];
  for i:=1 to N-1 do A[i]:=A[i+1];
  A[N] := c;
  { вывести полученный массив }
end.
```

Выбор нужных элементов

Задача – найти в массиве элементы, удовлетворяющие некоторому условию (например, отрицательные), и скопировать их в другой массив.

Примитивное решение:

```
const N = 5;
var i: integer;
 A, B: array[1..N]
                of integer;
begin
{ здесь заполнить массив A }
for i:=1 to N do
  if (A[i]<0) then
    B[i]:=A[i];
...
end.
```

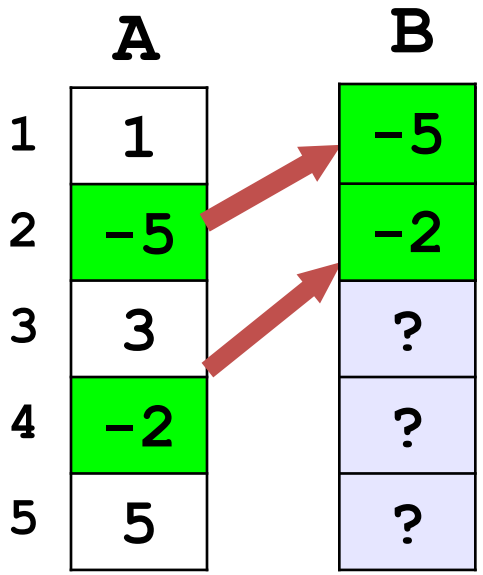


? Что плохо?

Выбор нужных элементов

Решение: ввести счетчик найденных элементов **count**, очередной элемент ставится на место **B[count]**.

```
count:=0;
for i:=1 to N do
  if (A[i]<0) then begin
    B[count]:=A[i];
    count:=count+1;
  end;
```



Как вывести массив B?

Примитивное решение:

```
writeln('Выбранные элементы:');
for i:=1 to N do
  write(B[i], ' ');
```

? Что плохо?

Правильное решение:

```
writeln('Выбранные элементы:');
for i:=1 to count do
  write(B[i], ' ');
```

Программирование на языке Паскаль
Часть II

Тема 4. Сортировка массивов

Сортировка

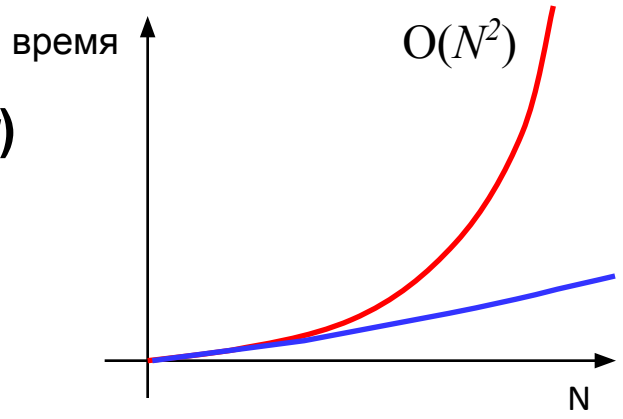
Сортировка – это расстановка элементов массива в заданном порядке (по возрастанию, убыванию, последней цифре, сумме делителей, …).

Задача: переставить элементы массива в порядке возрастания.

Алгоритмы:

сложность $O(N^2)$

- простые и понятные, но неэффективные для больших массивов
  - **метод пузырька**
  - **метод выбора**
- сложные, но эффективные
  - **«быстрая сортировка» (*Quick Sort*)**
  - **сортировка «кучей» (*Heap Sort*)**
  - **сортировка слиянием**
  - **пирамидальная сортировка**

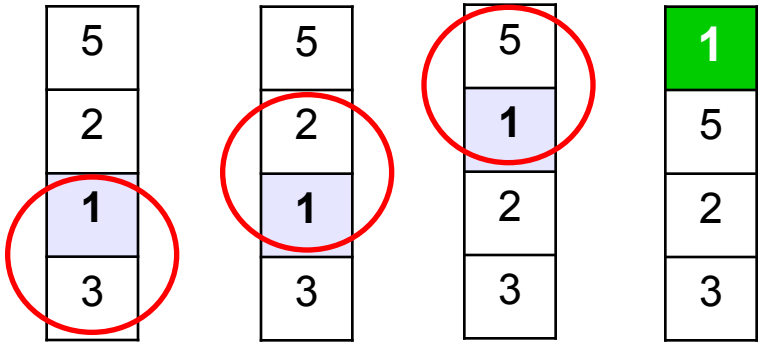


Метод пузырька

Идея – пузырек воздуха в стакане воды поднимается со дна вверх.

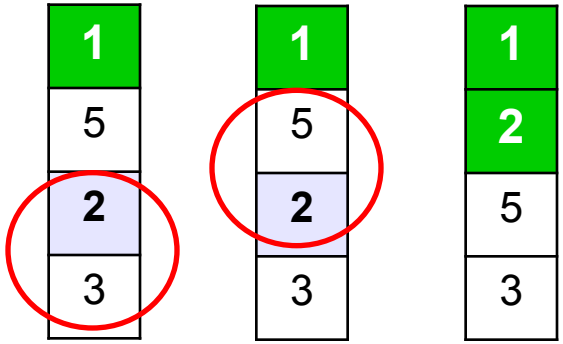
Для массивов – самый маленький («легкий» элемент перемещается вверх («всплывает»).

1-ый проход

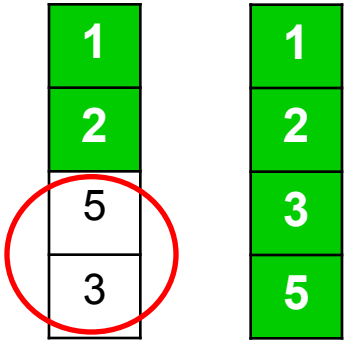


- начиная снизу, сравниваем два соседних элемента; если они стоят «неправильно», меняем их местами
- за 1 проход по массиву **один** элемент (самый маленький) становится на свое место

2-ой проход



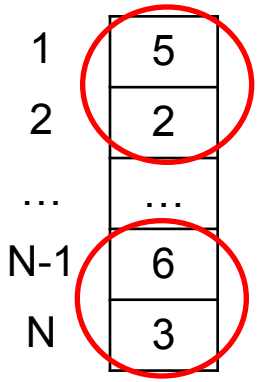
3-ий проход



Для сортировки массива из N элементов нужен N-1 проход (достаточно поставить на свои места N-1 элементов).

Программа

1-ый проход:



сравниваются пары

A[N-1] и A[N], A[N-2] и A[N-1]

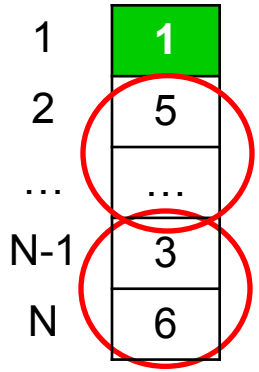
…

A[1] и A[2]

A[j] и A[j+1]

```
for j:=N-1 downto 1 do
  if A[j] > A[j+1] then begin
    c:=A[j]; A[j]:=A[j+1]; A[j+1]:=c;
  end;
```

2-ой проход



A[1] уже на своем месте!

```
for j:=N-1 downto 2 do
  if A[j] > A[j+1] then begin
    c:=A[j]; A[j]:=A[j+1]; A[j+1]:=c;
  end;
```

i-ый проход

```
for j:=N-1 downto i do
  ...
```

Программа

```
program qq;
const N = 10;
var A: array[1..N] of integer;
    i, j, c: integer;
begin
  { заполнить массив }
  { вывести исходный массив }
  for i:=1 to N-1 do begin
    for j:=N-1 downto i do
      if A[j] > A[j+1] then begin
        c := A[j];
        A[j] := A[j+1];
        A[j+1] := c;
      end;
  end;
  { вывести полученный массив }
end.
```

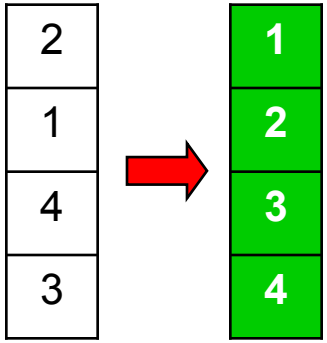
? Почему цикл по **i** до **N-1**?

элементы выше A[i] уже поставлены

Метод пузырька с флажком

Идея – если при выполнении метода пузырька не было обменов, массив уже отсортирован и остальные проходы не нужны.

Реализация: **переменная-флаг**, показывающая, был ли обмен; если она равна **`False`**, то выход.




```
var flag: boolean;
repeat
  flag := False; { сбросить флаг }
   for j:=N-1 downto 1 do
     if A[j] > A[j+1] then begin
       c := A[j];
       A[j] := A[j+1];
       A[j+1] := c;
       flag := True; { поднять флаг }
     end;
until not flag;  { выход при flag=False }
```

? Как улучшить?

Метод пузырька с флажком

```
  i := 0;
repeat
  i := i + 1;
  flag := False; { сбросить флаг }
  for j:=N-1 downto i do
    if A[j] > A[j+1] then begin
      c := A[j];
      A[j] := A[j+1];
      A[j+1] := c;
      flag := True; { поднять флаг }
    end;
until not flag;  { выход при flag=False }
```