Программирование на языке Паскаль
Часть II

Программирование на языке Паскаль Часть II

Тема 1. Массивы

Массивы

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

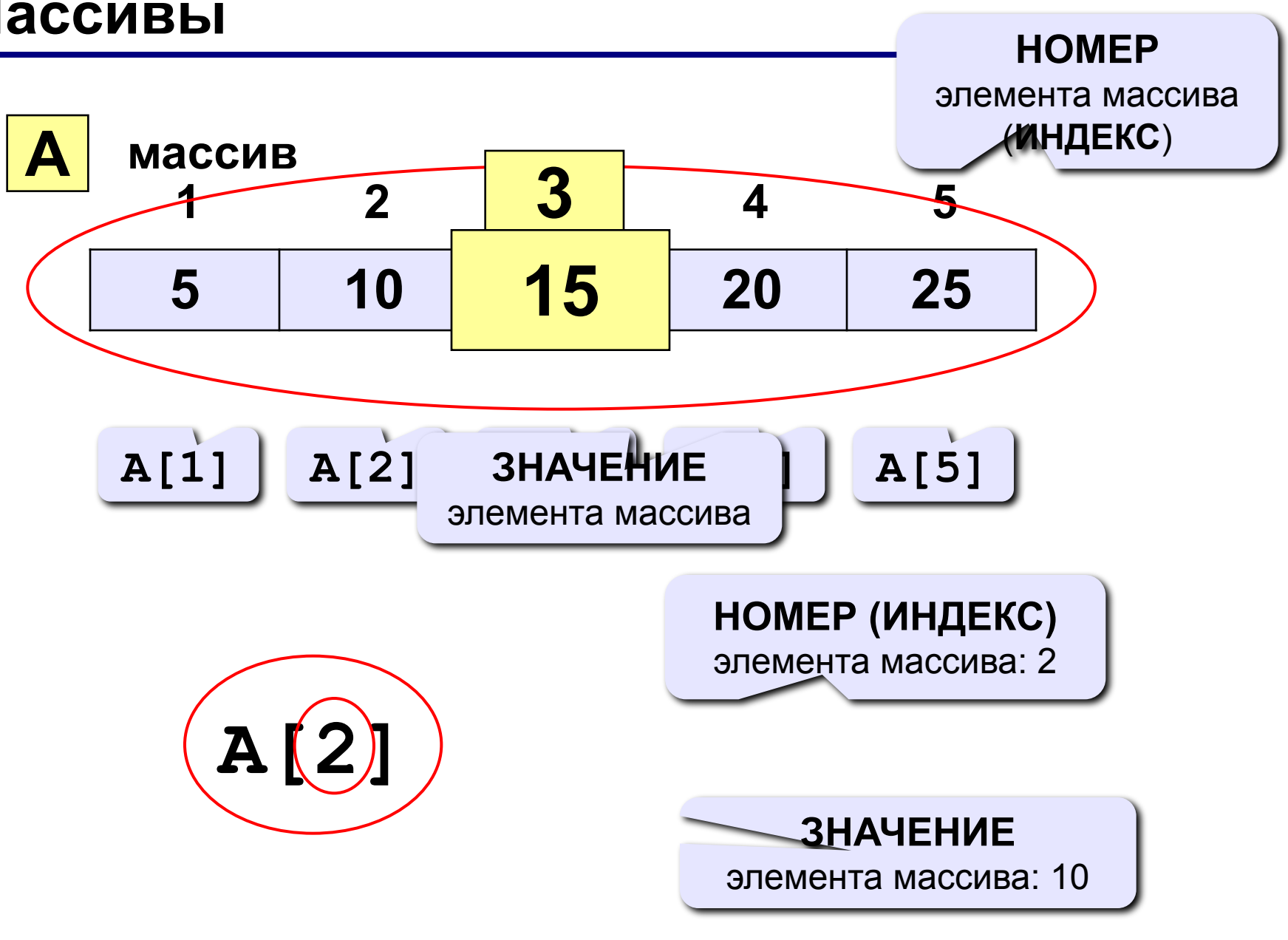
Особенности:

- 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]
A[5]
ЗНАЧЕНИЕ
элемента массива
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:


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

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[illegible]

```
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';
```

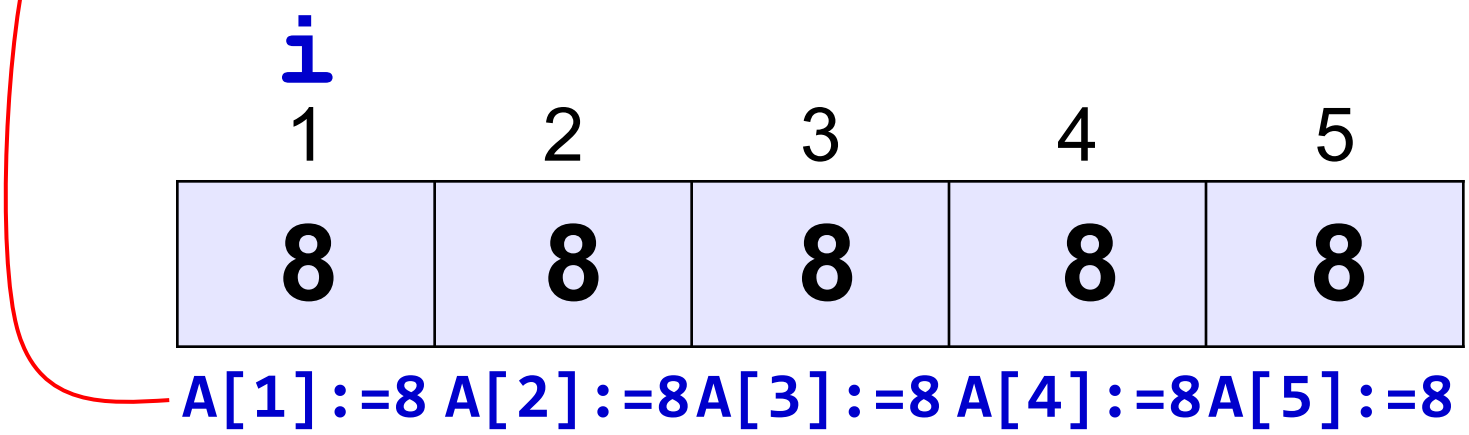
Заполнение массива

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;
```



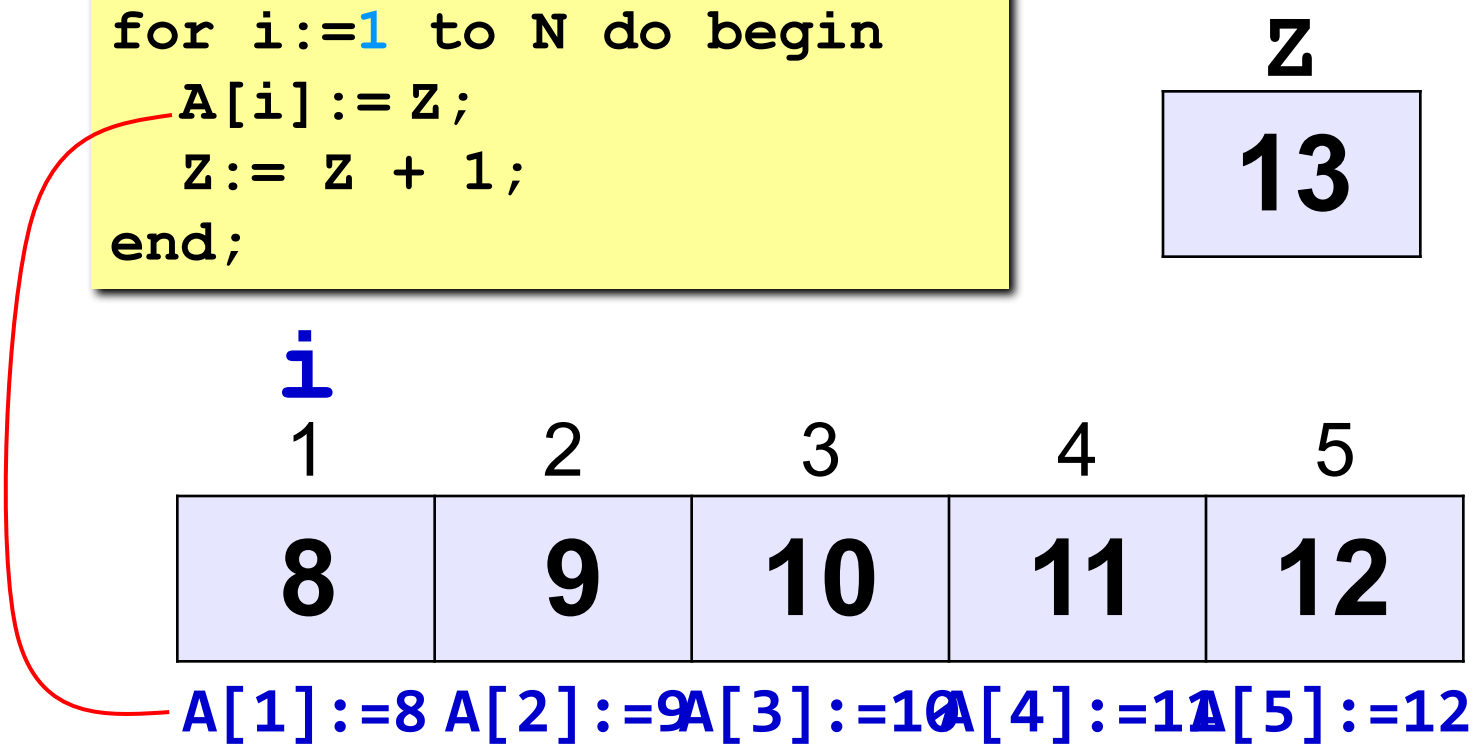
Заполнение массива

Объявление:

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

Заполнение последовательными числами:

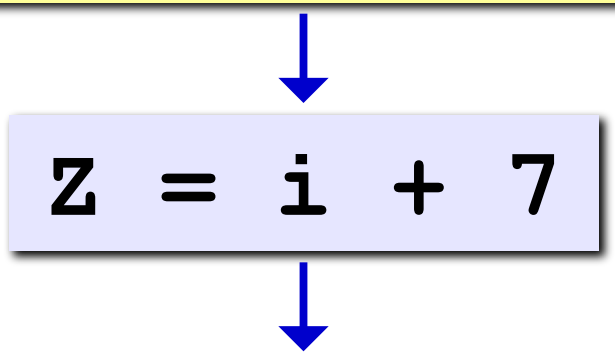
```
Z:= 8;
for i:=1 to N do begin
  A[i]:= Z;
  Z:= Z + 1;
end;
```



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?

Практикум: заполнение массива

«3»: 1. Заполните массив A нулями.

2. Заполните массив A первыми N натуральными числами, начиная с 1.

3. Заполните массив A первыми N натуральными числами, начиная с X (ввести X с клавиатуры).

«4»: 4. Заполните массив A первыми N натуральными числами, начиная с X (ввести X с клавиатуры) в обратном порядке (начиная с конца массива).

5. Заполнить массив A первыми N числами Фибоначчи. Первые два числа Фибоначчи равны единице, а каждое последующее число Фибоначчи вычисляется как сумма двух предыдущих.

«5»: 6. Заполните массив степенями числа 2, так чтобы последний элемент массива был равен 1, а каждый предыдущий был в 2 раза больше следующего. Например: 32 16 8 4 2 1

7. Заполните массив целыми числами, так чтобы средний элемент массива был равен X, слева от него элементы стоят по возрастанию, а справа – по убыванию (ввести X с клавиатуры). Соседние элементы отличаются на единицу. Например: 1 2 3 2 1.

Массивы

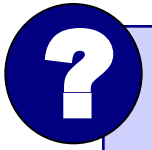
Объявление:

```
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
```

Задания

«3»: Ввести с клавиатуры массив из 5 элементов, умножить их на 2 и вывести на экран.

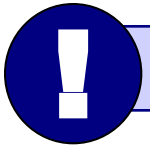
Пример:

```
Введите пять чисел:
4   15    3    10    14
Результат: 8 30 6 20 28
```

«4»: Ввести с клавиатуры массив из 5 элементов, найти среднее арифметическое всех элементов массива.

Пример:

```
Введите пять чисел:
4   15   3  10    14
среднее арифметическое 9.200
```



При изменении N остальная программа не должна изменяться!

Задания

«5»: Ввести с клавиатуры массив из 5 элементов, найти минимальный из них.

Пример:

```
Введите пять чисел:
4   15    3    10    14
минимальный элемент 3
```

Практикум: изменение элементов массива

«3»:

1. Увеличить все элементы массива A на 1.
2. Умножить все элементы массива A на 2.
3. Возвести в квадрат все элементы массива A.

«4»:

4. Увеличить на 4 все элементы в первой половине массива A (считать, что в массиве чётное число элементов).
5. Разделить на 2 все элементы массива A, кроме первого и последнего (считать, что в массиве есть, по крайней мере, два элемента и все элементы чётные).

«5»:

6. Умножить на 3 все элементы во второй половине массива A (считать, что в массиве чётное число элементов).
7. Найти среднее арифметическое всех элементов массива A.

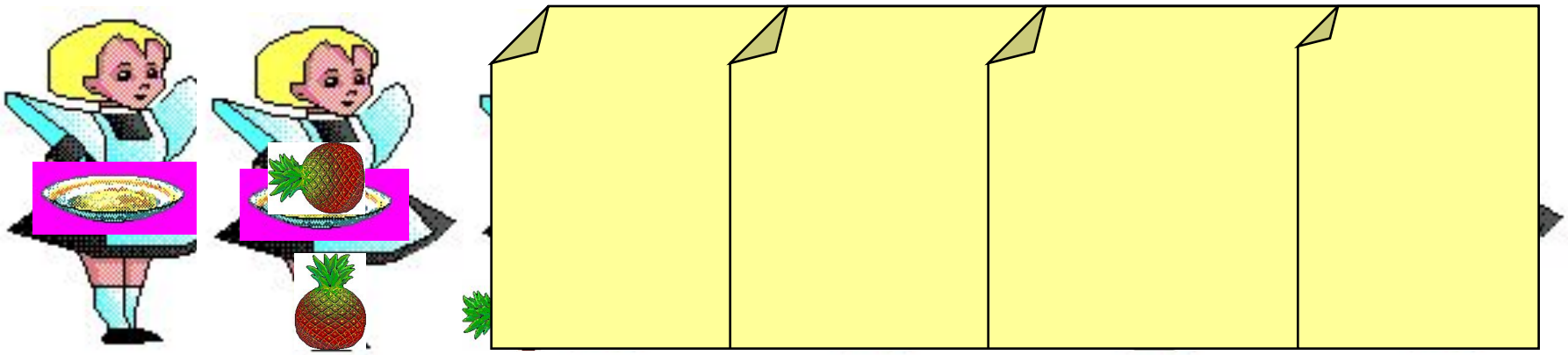
Программирование на языке Паскаль
Часть II

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.
```

Задания

«3»: Ввести с клавиатуры массив из 5 элементов, найти в нем минимальный элемент и его номер.

Пример:

```
Исходный массив:
4   -5   10  -10  5
мимимальный A[4]=-10
```

«4»: Ввести с клавиатуры массив из 5 элементов, найти в нем максимальный и минимальный элементы и их номера.

Пример:

```
Исходный массив:
4   -5   10  -10  5
максимальный A[3]=10
минимальный  A[4]=-10
```

Задания

«5»: **Ввести с клавиатуры массив из 5 элементов, найти в нем два максимальных элемента и их номера.**

Пример:

```
Исходный массив:
4   -5   10  -10  5
максимальные A[3]=10, A[5]=5
```

Практикум: максимум/минимум

«3»:

1. Найти максимальное значение среди всех элементов массива.
2. Найти минимальное значение среди всех элементов массива.
3. Найти минимальное и максимальное значения среди всех элементов массива.

«4»:

4. Найти номер минимального элемента массива.
5. Найти номера минимального и максимального элементов массива.

«5»:

6. Найти два максимальных элемента массива.
7. Найти номера двух минимальных элементов массива.

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

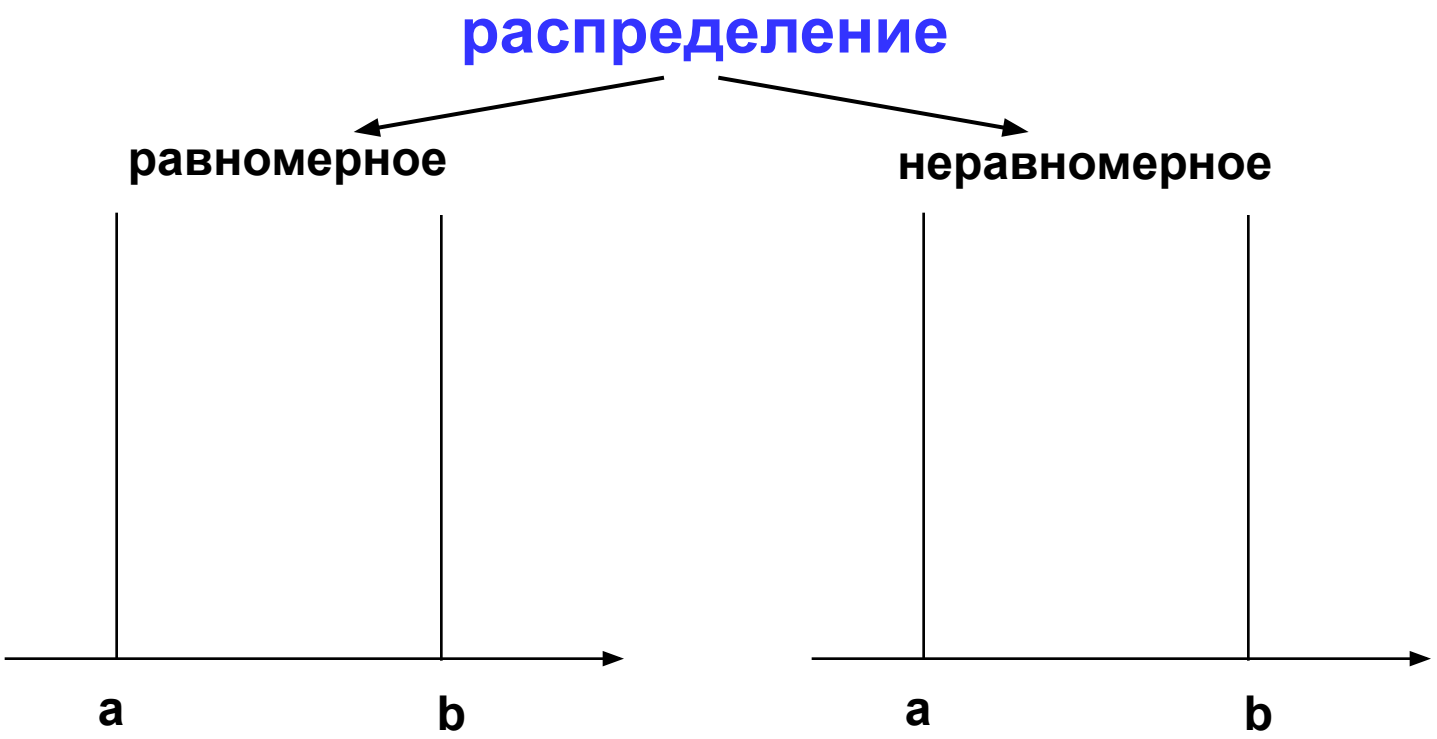
318458191041

209938992481

малый период (последовательность повторяется через 10^6 чисел)

Распределение случайных чисел

Модель: снежинки падают на отрезок [a,b]

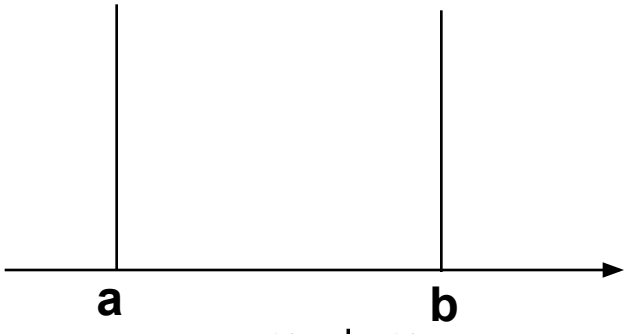


Сколько может быть разных распределений?

Распределение случайных чисел

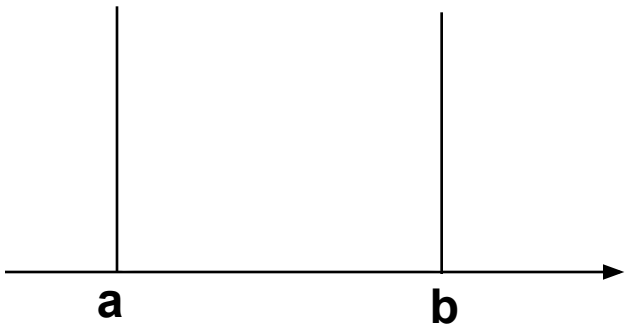
Особенности:

- 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



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

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

Какой отрезок?

[0,N-1]

[a,a+N-1]

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

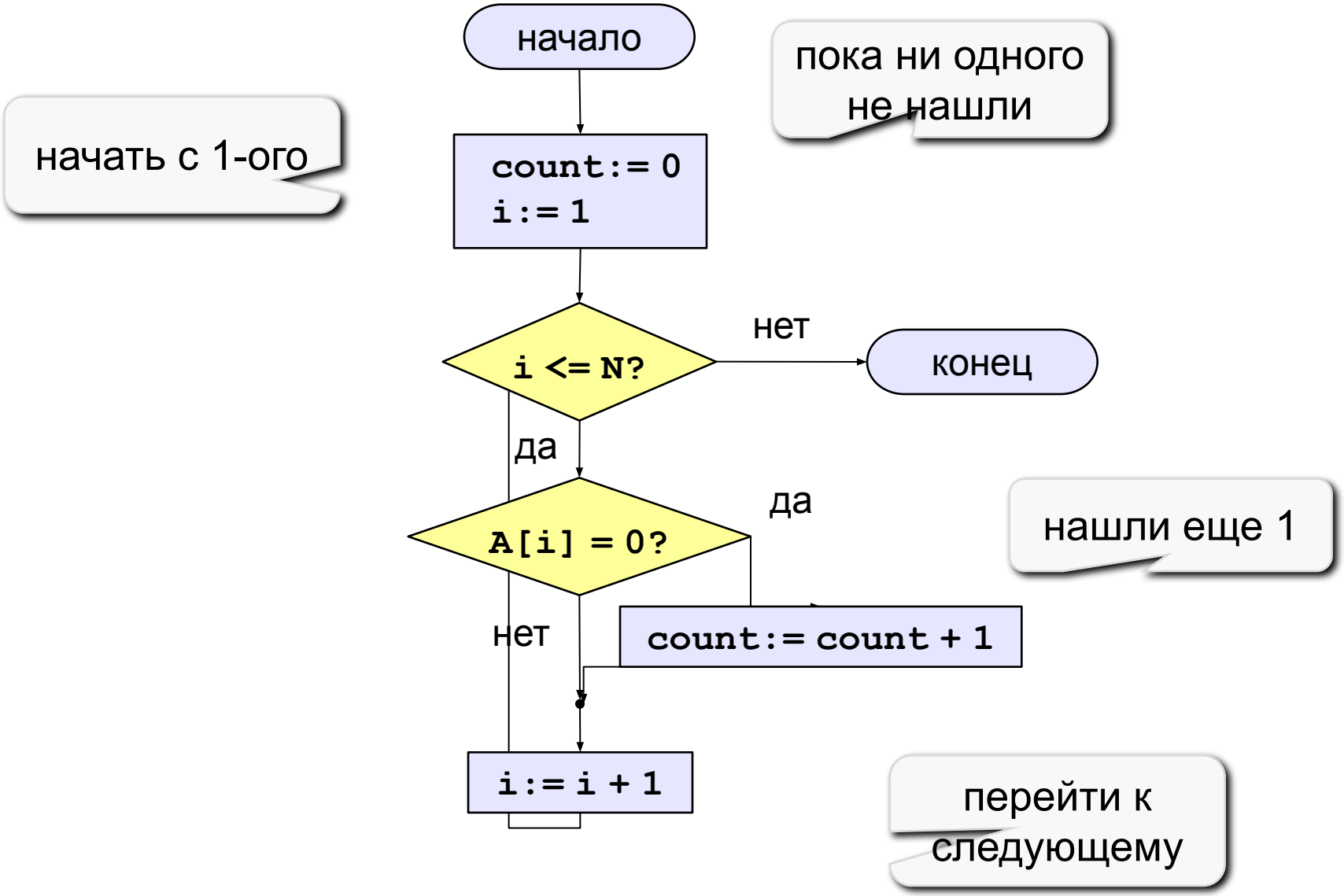
Задача: заполнить массив случайными числами в интервале `[-1,1]` и подсчитать количество нулевых элементов.

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

Задания

«3»: Заполнить массив случайными числами в интервале `[-2,2]` и подсчитать количество положительных элементов.

«4»: Заполнить массив случайными числами в интервале `[20,100]` и подсчитать отдельно число чётных и нечётных элементов.

«5»: Заполнить массив случайными числами в интервале `[1000,2000]` и подсчитать число элементов, у которых вторая с конца цифра – четная.

Практикум: подсчёт элементов массива

«3»:

1. Определите, сколько элементов массива A равны 1.
2. Определите, сколько элементов массива A равны заданному значению X.
3. Определите количество положительных элементов массива A.

«4»:

4. Определите количество чётных и нечётных элементов массива A.
5. Определите, количество чётных положительных элементов массива A.

«5»:

6. Найти количество элементов массива, в десятичной записи которых предпоследняя цифра (число десятков) – 5.
7. Найти количество элементов массива, в десятичной записи которых последняя и предпоследняя цифры одинаковые.

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:

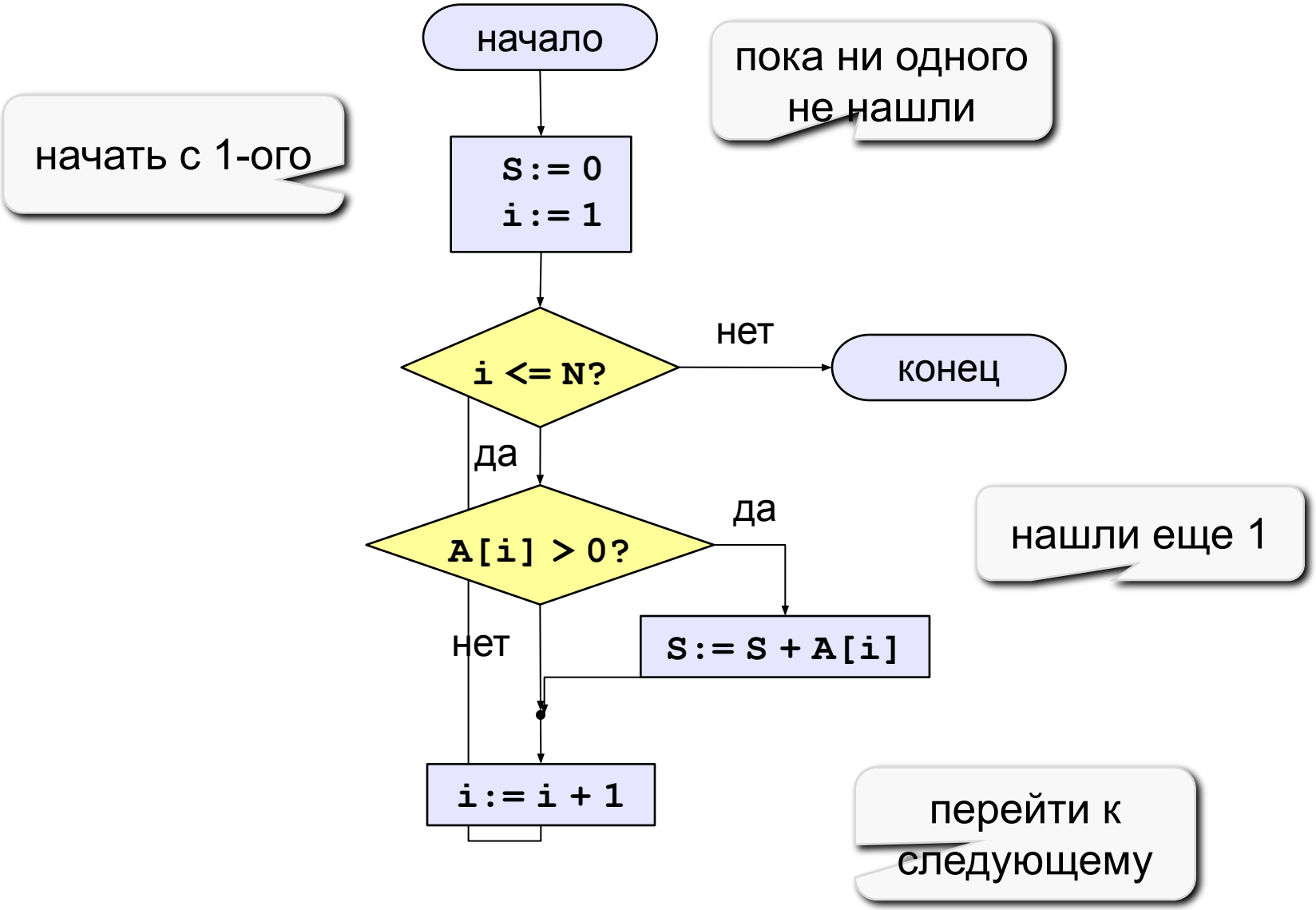
если очередной элемент > 0,

то добавить к сумме этот элемент

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

3)вывести значение суммы

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

Задания

«3»: Заполнить массив из 10 элементов случайными числами в интервале `[-10,10]` и подсчитать сумму всех отрицательных элементов.

«4»: Заполнить массив из 10 элементов случайными числами в интервале `[0,100]` и подсчитать среднее значение всех элементов, которые <50.

«5»: Заполнить массив из 10 элементов случайными числами в интервале `[10,12]` и найти длину самой длинной последовательности стоящих рядом одинаковых элементов.

Пример:

```
Исходный массив:
10  10  11  12  12  12  10  11  11  12
Длина последовательности: 3
```

Практикум: суммы, прозведения…

«3»: 1. Вычислить сумму всех элементов массива A.

2. Вычислить сумму отрицательных элементов массива A.

3. Вычислить сумму всех элементов массива A, которые делятся на 3.

«4»: 4. Вычислить среднее арифметическое всех элементов массива A, которые меньше, чем 50.

5. Вычислить произведение всех чётных положительных элементов массива A.

«5»:

6. Найти сумму всех элементов массива A, у которых число десятков (вторая с конца цифра десятичной записи) больше, чем число единиц.

7. Все элементы массива A - трёхзначные числа. Найти сумму всех элементов массива A, в десятичной записи которых все цифры одинаковые.

Поиск в массиве

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

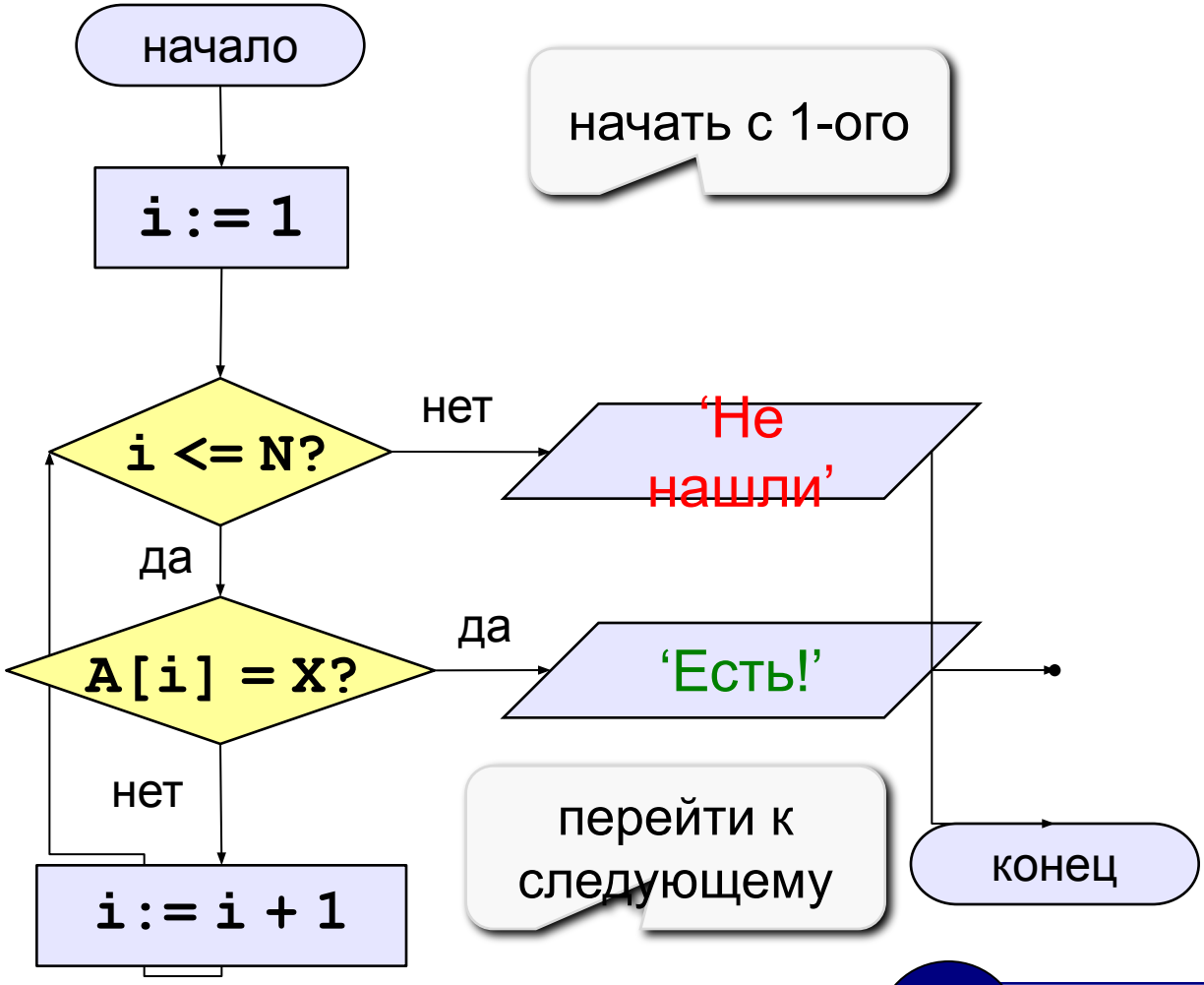
Пример: если в классе ученик с фамилией Пупкин?

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.
```

Задания

«3»: Заполнить массив из 10 элементов случайными числами в интервале `[10..20]` и найти элемент, равный X.

Пример:

```
Исходный массив:
13  10  18  12  20  11  13  14  15  20
Что ищем? 20
A[5] = 20
```

«4»: Заполнить массив из 10 элементов случайными числами в интервале `[0..4]` и вывести номера всех элементов, равных X.

Пример:

```
Исходный массив:
4  0  1  2  0  1  3  4  1  0
Что ищем? 0
A[2], A[5], A[10]
```

Задания

«5»: Заполнить массив из 10 элементов случайными числами в интервале `[0..4]` и определить, есть ли в нем одинаковые соседние элементы.

Пример:

```
Исходный массив:
4  0  1  2  0  1  3  1  1  0
Ответ: есть
```

Практикум: суммы, прозведения…

«3»: 1. Определите в массиве A номер первого элемента, равного X.

2. Определите номер первого элемента, равного X, в первой половине массива A (массив имеет чётное число элементов).

3. Определите номер первого элемента, равного X, во второй половине массива A (массив имеет чётное число элементов).

«4»: 4. Определите номер последнего элемента, равного X, во второй половине массива A (массив имеет чётное число элементов).

5. Определите, сколько есть элементов, равных X, в первой половине массива A (массив имеет чётное число элементов).

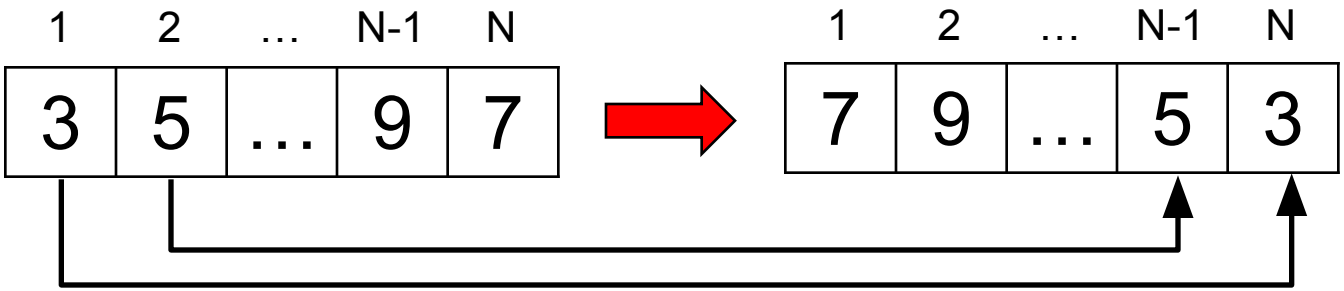
«5»:

6. Определите, сколько в массиве A пар соседних элементов, значения которых одинаковы и равны заданному X.

7. Горка – это три стоящих подряд элемента массива A, из которых средний ("вершина") имеет наибольшее значение, а два крайних - меньше него. Найти количество "горок" в массиве A, в которых значение среднего элемента равно X..

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?

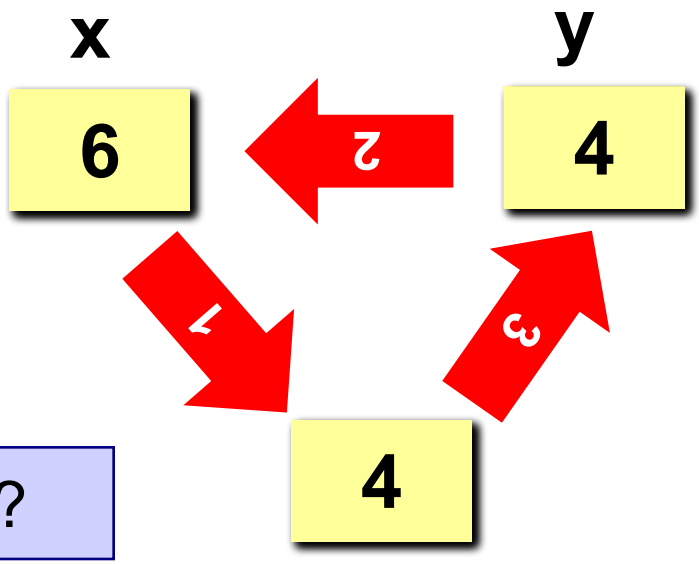
Как переставить элементы?

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.
```

Задания

«3»: Заполнить массив из 10 элементов случайными числами в интервале `[-10..10]` и сделать реверс всех элементов, кроме последнего.

Пример:

```
Исходный массив:
-5  3  10  -4  -6  8 -10  1  0  4
Результат:
 0  1 -10   8  -6 -4  10  3 -5  4
```

«4»: Заполнить массив из 10 элементов случайными числами в интервале `[-10..10]` и сделать реверс всех элементов, кроме первого.

Пример:

```
Исходный массив:
4 -5  3  10 -4  -6  8 -10  1  0
Результат:
4  0  1 -10  8  -6 -4  10  3 -5
```

Задания

«5»: Заполнить массив из 10 элементов случайными числами в интервале `[-10..10]` и сделать реверс отдельно для 1-ой и 2-ой половин массива.

Пример:

```
Исходный массив:
4   -5   3  10  -4  -6  8  -10  1  0
Результат:
-4  10   3  -5   4   0  1  -10  8 -6
```

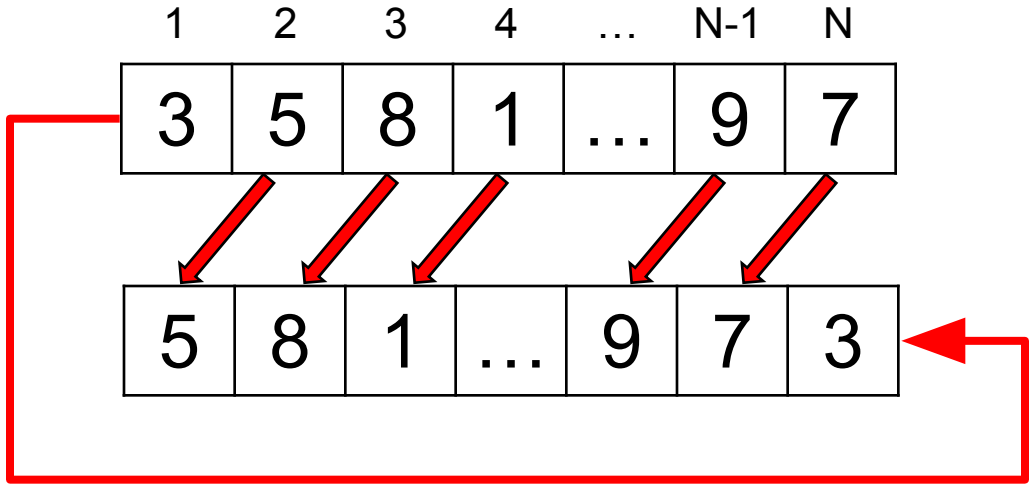
«6»: Заполнить массив из 12 элементов случайными числами в интервале `[-12..12]` и выполнить реверс для каждой трети массива.

Пример:

```
Исходный массив:
4   -5   3  10  -4  -6   8 -10  1  0  5  7
Результат:
10   3  -5   4 -10   8  -6  -4  7  5  0  1
```

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.
```

Задания

«3»: Заполнить массив из 10 элементов случайными числами в интервале `[-10..10]` и выполнить циклический сдвиг влево *без первого элемента*.

Пример:

```
Исходный массив:
 4  -5   3  10  -4  -6   8 -10  1  0
Результат:
 4   3  10  -4  -6   8 -10   1  0 -5
```

«4»: Заполнить массив из 10 элементов случайными числами в интервале `[-10..10]` и выполнить циклический сдвиг ВПРАВО.

Пример:

```
Исходный массив:
 4   -5   3  10  -4  -6  8  -10  1  0
Результат:
 0   4   -5   3  10  -4  -6  8  -10  1
```

Задания

«5»: Заполнить массив из 12 элементов случайными числами в интервале `[-12..12]` и выполнить циклический сдвиг ВПРАВО на 4 элемента.

Пример:

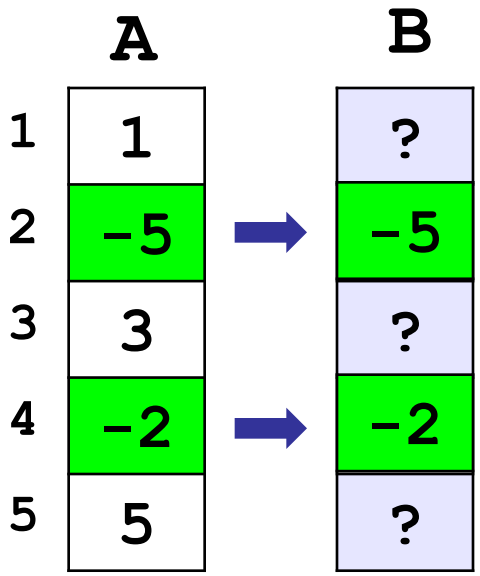
```
Исходный массив:
4  -5   3  10  -4  -6  8  -10 | 1   0   5   7
Результат:
1   0   5   7 | 4  -5  3   10  -4  -6   8 -10
```

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.
```

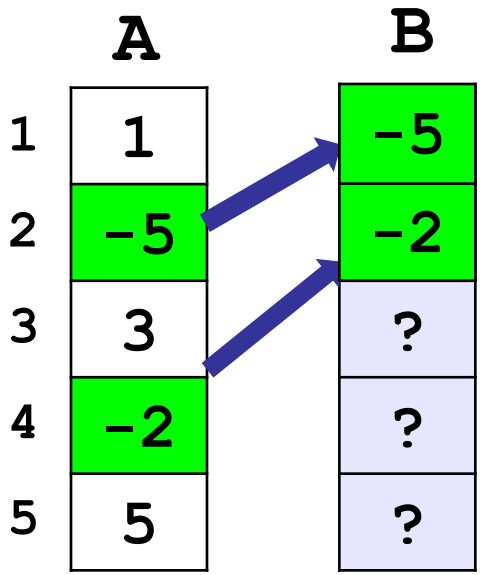


Что плохо?

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], ' ');
```

Задания

«3»: Заполнить массив случайными числами в интервале [-10,10] и записать в другой массив все положительные числа.

Пример:

```
Исходный массив:
0  -5   3  7  -8
Положительные числа:
3  7
```

«4»: Заполнить массив случайными числами в интервале [20,100] и записать в другой массив все числа, которые оканчиваются на 0.

Пример:

```
Исходный массив:
40   57   30  71  84
Заканчиваются на 0:
40 30
```

Задания

«5»: Заполнить массив случайными числами и выделить в другой массив все числа, которые встречаются более одного раза.

Пример:

```
Исходный массив:
4  1  2  1  11  2  34
Результат:
1  2
```

Программирование на языке Паскаль
Часть II

Тема 4. Сортировка массивов

Сортировка

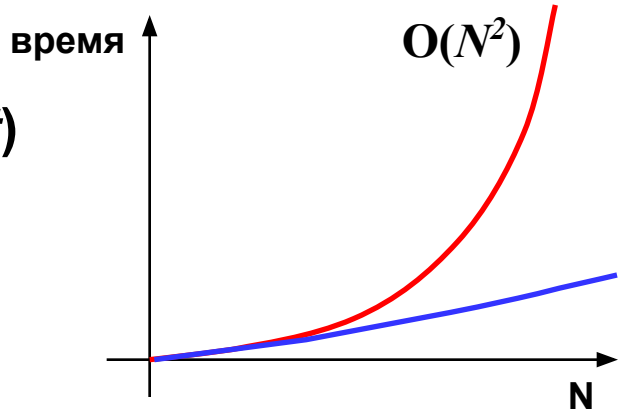
Сортировка – это расстановка элементов массива в заданном порядке (по возрастанию, убыванию, последней цифре, сумме делителей, …).

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**

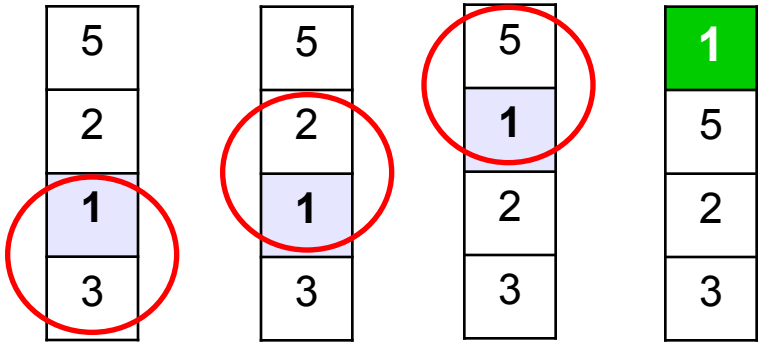


Метод пузырька

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

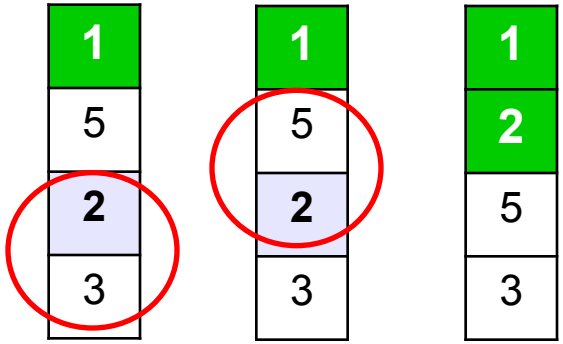
Для массивов – самый маленький («легкий» элемент перемещается вверх («всплывает»).

1-ый проход

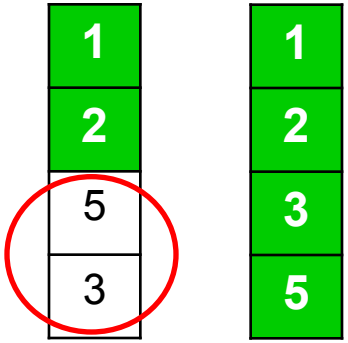


- начиная снизу, сравниваем два соседних элемента; если они стоят «неправильно», меняем их местами
- за 1 проход по массиву **один** элемент (самый маленький) становится на свое место

2-ой проход



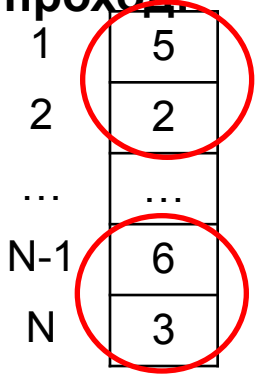
3-ий проход



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]`

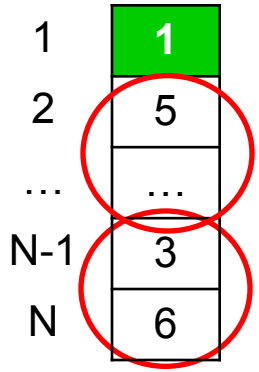
…

`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

Задания

«3»: Заполнить массив из 10 элементов случайными числами в интервале `[-10..10]` и отсортировать его по убыванию.

Пример:

```
Исходный массив:
4  5  -8  3  -7  -5  3  1  0  9
Результат:
9  5  4  3  3  1  0  -5  -7  -8
```

«4»: Заполнить массив из 10 элементов случайными числами в интервале `[0..100]` и отсортировать его по последней цифре.

Пример:

```
Исходный массив:
14  25  13  30  76  58  32  11  41  97
Результат:
30  11  41  32  13  14  25  76  97  58
```

Задания

«5»: **Заполнить массив из 10 элементов случайными числами в интервале `[0..100]` и отсортировать первую половину по возрастанию, а вторую – по убыванию.**

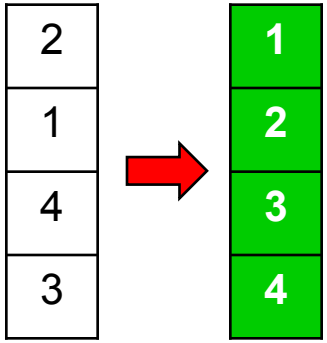
Пример:

```
Исходный массив:
14  25  13  30  76  58  32  11  41  97
Результат:
13  14  25  30  76  97  58  41  32  11
```

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?

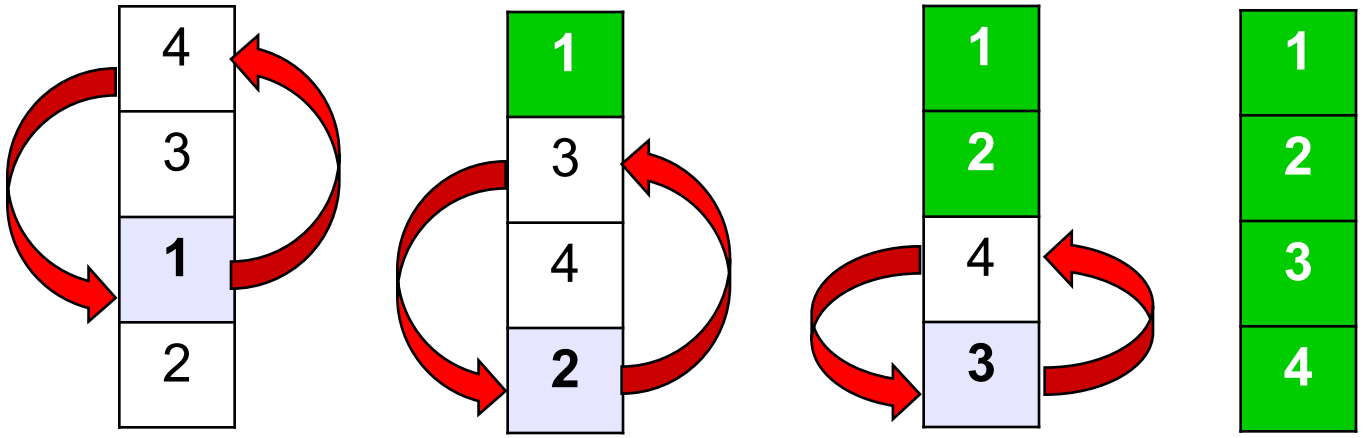
Метод пузырька с флажком

```
  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 }
```

Метод выбора

Идея:

- найти минимальный элемент и поставить на первое место (поменять местами с **`A[1]`**)
- **из оставшихся** найти минимальный элемент и поставить на второе место (поменять местами с **`A[2]`**), и т.д.



Метод выбора

нужно N-1 проходов

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

поиск минимального от A[i] до A[N]

если нужно, переставляем

? Можно ли убрать if?

Задания

«3»: Заполнить массив из 10 элементов случайными числами в интервале [0..99] и отсортировать его по убыванию последней цифры.

Пример:

```
Исходный массив:
14  25  13  12  76  58  21  87  10  98
Результат:
98  58  87  76  25  14  13  12  21  10
```

«4»: Заполнить массив из 10 элементов случайными числами в интервале [0..99] и отсортировать его по возрастанию суммы цифр (*подсказка: их всего две*)**.**

Пример:

```
Исходный массив:
14  25  13  12  76  58  21  87  10  98
Результат:
10  21  12  13  14  25  76  58  87  98
```

Задания

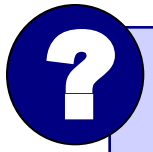
«5»: **Заполнить массив из 10 элементов случайными числами в интервале [0..100] и отсортировать первую половину по возрастанию, а вторую – по убыванию.**

Пример:

```
Исходный массив:
14  25  13  30  76  58  32  11  41  97
Результат:
13  14  25  30  76  97  58  41  32  11
```

«Быстрая сортировка» *(Quick Sort)*

Идея – более эффективно переставлять элементы, расположенные дальше друг от друга.

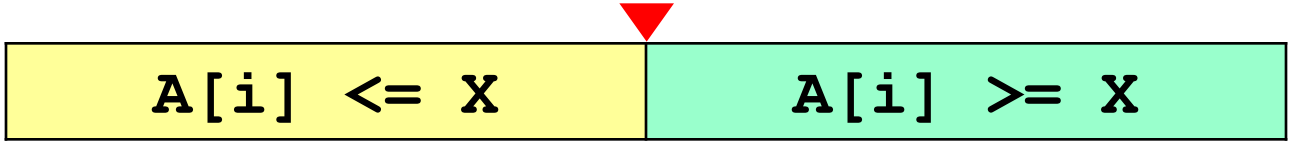


Сколько перестановок нужно, если массив отсортирован по убыванию, а надо – по возрастанию?

`N div 2`

1 шаг: выбрать некоторый элемент массива X

2 шаг: переставить элементы так:



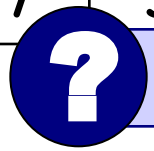
при сортировке элементы не покидают « свою область»!

3 шаг: так же отсортировать две получившиеся области

Разделяй и властвуй (англ. *divide and conquer*)

«Быстрая сортировка» *(Quick Sort)*

78	6	82	67	55	44	34



Как лучше выбрать **X**?

Медиана – такое значение X, что слева и справа от него в отсортированном массиве стоит одинаковое число элементов (*для этого надо отсортировать массив…*).

Разделение:

1)выбрать средний элемент массива (**X=67**)

78	6	82	67	55	44	34

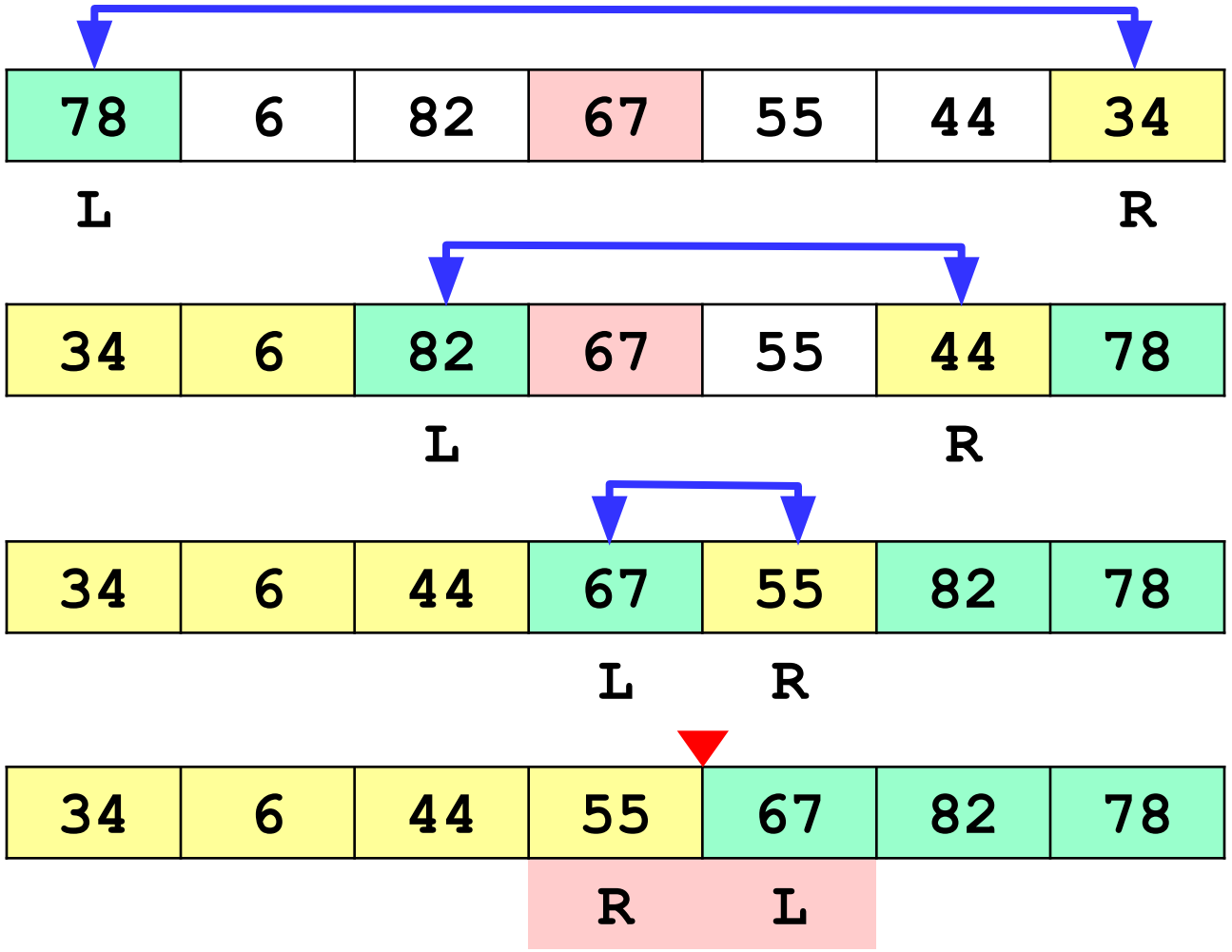
1)установить **L:=1**, **R:=N**

2)увеличивая **L**, найти первый элемент **A[L]**, который **>= X** (должен стоять справа)

3)уменьшая **R**, найти первый элемент **A[R]**, который **<= X** (должен стоять слева)

4)если **L<=R**, поменять местами **A[L]** и **A[R]** и перейти к п. 3

«Быстрая сортировка» *(Quick Sort)*



L > R : разделение закончено

«Быстрая сортировка» *(Quick Sort)*

```
procedure QSort ( first, last: integer);
var L, R, c, X: integer;
begin
  if first < last then begin
    X:=A[(first+last) div 2];
    L:=first; R:=last;
    while L <= R do begin
      while A[L]<X do L:=L+1;
      while A[R]>X do R:=R-1;
      if L<=R then begin
        c:=A[L]; A[L]:=A[R]; A[R]:=c;
        L:=L+1; R:=R-1;
      end;
    end;
    QSort(first, R);   QSort(L, last);
  end;
end.
```

ограничение рекурсии

разделение

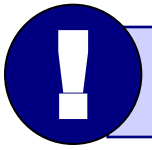
обмен

двигаемся дальше

сортируем две части

«Быстрая сортировка» *(Quick Sort)*

```
program qq;
const N = 10;
var A: array[1..N] of integer;
procedure QSort ( first, last: integer);
...
begin
   { заполнить массив }
   { вывести исходный массив на экран }
   Qsort ( 1, N ); { сортировка }
   { вывести результат }
end.
```



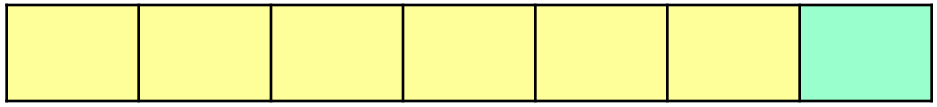
Сложность (в среднем) $O(N \log N)$!

Количество перестановок (случайные данные)

N	*QuickSort* $O(N \log N)$	*«пузырек»* $O(N^2)$
10	11	24
100	184	2263
200	426	9055
500	1346	63529
1000	3074	248547

От чего зависит скорость?

Как хуже всего выбирать X?



$O(N^2)$

Задания

«3»: Заполнить массив из 10 элементов случайными числами в интервале `[-50..50]` и отсортировать его с помощью алгоритма быстрой сортировки.

«4»: Заполнить массив из 10 элементов случайными числами в интервале `[-50..50]` и отсортировать его по убыванию с помощью алгоритма быстрой сортировки.

«5»: Заполнить массив из 500 элементов случайными числами в интервале `[0..100]`. Отсортировать его по возрастанию двумя способами – методом «пузырька» и методом «быстрой сортировки» . Вывести на экран число перестановок элементов массива в том и в другом случае. Массив выводить на экран не нужно.

Программирование на языке Паскаль
Часть II

Тема 5. Двоичный поиск

Поиск в массиве

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

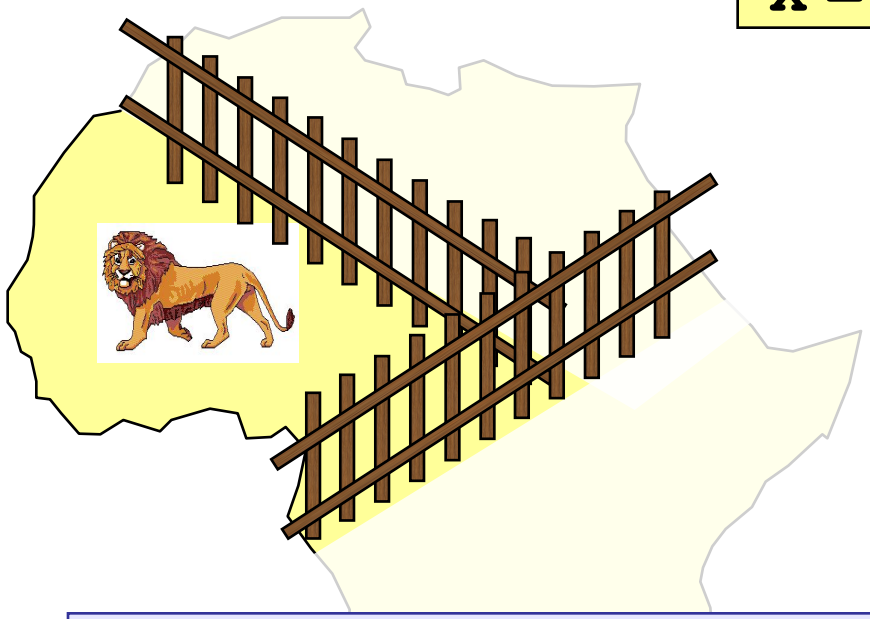
Решение: для произвольного массива: **линейный поиск** (перебор)

недостаток: **низкая скорость**

Как ускорить? – заранее подготовить массив для поиска

- как именно подготовить?
- как использовать «подготовленный массив»?

Двоичный поиск



X = 7

1	1	1	1
2	2	2	2
3	3	3	3
4		4	4
5		5	5
6		6	6
7		7	7
8			8
9			9
10			10
11			11
12			12
13			13
14			14
15			15
16			16

X < 8

X > 4

X > 6

1. Выбрать средний элемент A[c] и сравнить с X.
2. Если X = A[c], нашли (выход).
3. Если X < A[c], искать дальше в первой половине.
4. Если X > A[c], искать дальше во второй половине.

Двоичный поиск


```
nX := 0;
L := 1; R := N; {границы: ищем от A[1] до A[N] }
while R >= L do begin
  c := (R + L) div 2;
  if X = A[c] then begin
    nX := c;
    R := L - 1; { break; }
  end;
  if X < A[c] then R := c - 1;
  if X > A[c] then L := c + 1;
end;
if nX < 1 then writeln('Не нашли...')
else           writeln('A[', nX, ']=', X);
```

номер среднего элемента

нашли

выйти из цикла

сдвигаем границы

Почему нельзя `while R > L do begin … end;` ?

Сравнение методов поиска

	Линейный	Двоичный
подготовка	нет	отсортировать
	число шагов	
N = 2	2	2
N = 16	16	5
N = 1024	1024	11
N= 1048576	1048576	21
N	$\le N$	$\le \log_2 N+1$

Задания

«3»: Написать программу, которая сортирует массив по возрастанию и ищет в нем элемент, равный X (это число вводится с клавиатуры). Использовать двоичный поиск.

«4»: Написать программу, которая сортирует массив ПО УБЫВАНИЮ и ищет в нем элемент, равный X (это число вводится с клавиатуры). Использовать двоичный поиск.

«5»: Написать программу, которая считает среднее число шагов в двоичном поиске для массива из 32 элементов в интервале `[0,100]`. Для поиска использовать 1000 случайных чисел в этом же интервале.

Программирование на языке Паскаль
Часть II

Тема 6. Символьные строки

Чем плох массив символов?

Это массив символов:

```
var B: array[1..N] of char;
```

- каждый символ – отдельный объект;
- массив имеет длину N, которая задана при объявлении

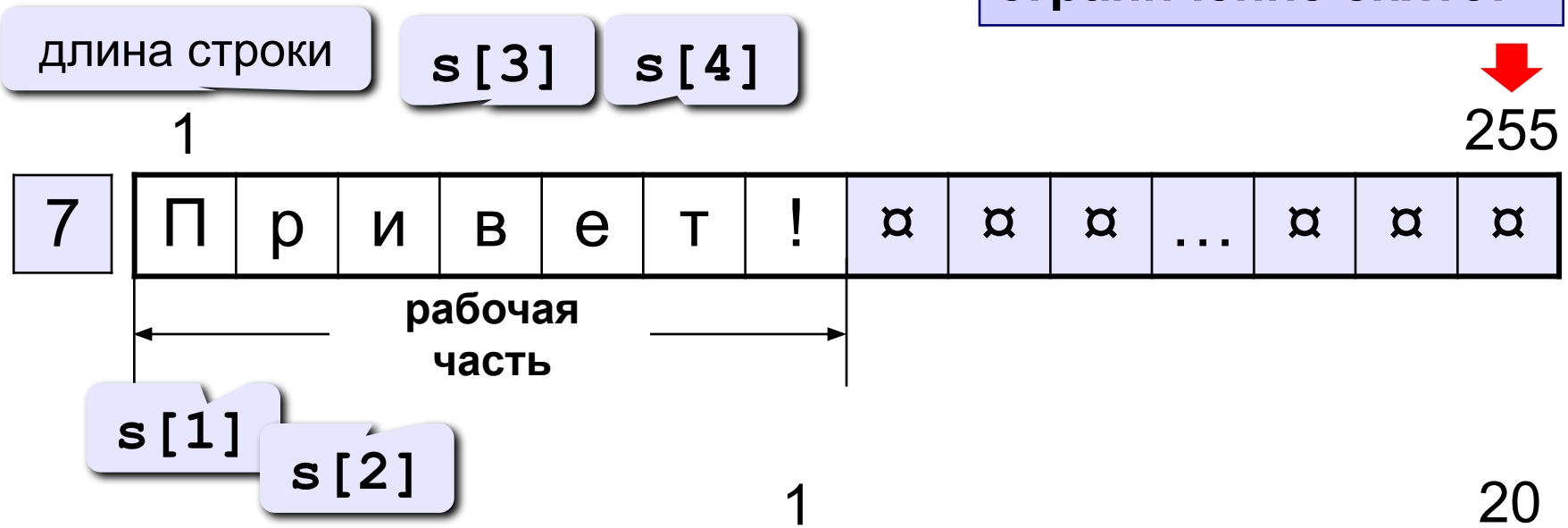
Что нужно:

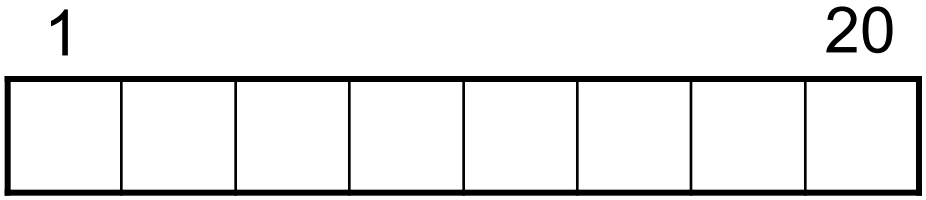
- обрабатывать последовательность символов как единое целое
- строка должна иметь переменную длину

Символьные строки

```
var s: string;
```

В *Delphi* это ограничение снято!






```
var s: string[20];
```

Длина строки:

```
var n: integer;
```

```
n := length ( s );
```

Символьные строки

Задача: ввести строку с клавиатуры и заменить все буквы «а» на буквы «б».

```
program qq;
var s: string;
    i: integer;
begin
   writeln('Введите строку');
   readln(s);
   for i:=1 to Length(s) do
      if s[i] = 'а' then s[i] := 'б';
   writeln(s);
end.
```

ввод строки

длина строки

вывод строки

Задания

«3»: Ввести символьную строку и заменить все буквы «а» на буквы «б», как заглавные, так и строчные.

Пример:

```
Введите строку:
ааббссААББСС
Результат:
ббббссББББСС
```

«4»: Ввести символьную строку и заменить все буквы «а» на буквы «б» и наоборот, как заглавные, так и строчные.

Пример:

```
Введите строку:
ааббссААББСС
Результат:
ббаассББААСС
```

Задания

«5»: Ввести символьную строку и проверить, является ли она палиндромом (палиндром читается одинаково в обоих направлениях).

Пример:

```
Введите строку:
АБВГДЕ
Результат:
Не палиндром.
```

Пример:

```
Введите строку:
КАЗАК
Результат:
Палиндром.
```

Операции со строками

```
var s, s1, s2: string;
```

Запись нового значения:

```
s := 'Вася';
```

Объединение: добавить одну строку в конец другой.

```
s1 := 'Привет';
s2 := 'Вася';
s := s1 + ', ' + s2 + '!';
```

`'Привет, Вася!'`

Подстрока: выделить часть строки в другую строку.

```
s := '123456789';

s1 := Copy ( s, 3, 6 );
s2 := Copy ( s1, 2, 3 );
```

с 3-его символа

6 штук

`'345678'`

`'456'`

Удаление и вставка

Удаление части строки:

```
s := '123456789';
Delete ( s, 3, 6 );
```

6 штук

'123456789'

'129'

строка меняется!

с 3-его символа

Вставка в строку:

```
s := '123456789';
Insert ( 'ABC', s, 3 );

Insert ( 'Q', s, 5 );
```

начиная с 3-его символа

'12ABC3456789'

что вставляем

куда вставляем

'12ABQC3456789'

Поиск в строке

Поиск в строке:

s[3]

```
var n: integer;
```

```
s := 'Здесь был Вася.';
n := Pos ( 'e', s );
if n > 0 then
     writeln('Буква e - это s[', n, ']')
else writeln('Не нашли');
n := Pos ( 'Вася', s );
s1 := Copy ( s, n, 4 );
```

3

n = 11

Особенности:

- функция возвращает номер символа, с которого начинается образец в строке
- если слова нет, возвращается 0
- поиск с начала (находится **первое** слово)

Примеры

```
s := 'Вася Петя Митя';
n := Pos ( 'Петя', s );
Delete ( s, n, 4 );
Insert ( 'Лена', s, n );
```

```
6
'Вася  Митя'
'Вася Лена Митя'
```

```
s := 'Вася Петя Митя';
n := length ( s );
s1 := Copy ( s, 1, 4 );
s2 := Copy ( s, 11, 4 );
s3 := Copy ( s, 6, 4 );
s := s3 + s1 + s2;
n := length ( s );
```

```
14
'Вася'
'Митя'
'Петя'
'ПетяВасяМитя'
12
```

Пример решения задачи

Задача: Ввести имя, отчество и фамилию. Преобразовать их к формату «фамилия-инициалы».

Пример:

```
Введите имя, фамилию и отчество:
Василий Алибабаевич Хрюндиков
Результат:
Хрюндиков В.А.
```

Алгоритм:

- найти первый пробел и выделить имя
- удалить имя с пробелом из основной строки
- найти первый пробел и выделить отчество
- удалить отчество с пробелом из основной строки
- «сцепить» фамилию, первые буквы имени и фамилии, точки, пробелы…

Программа

```
program qq;
var s, name, otch: string;
    n: integer;
begin
   writeln('Введите имя, отчество и фамилию');
   readln(s);
   n := Pos(' ', s);
   name := Copy(s, 1, n-1); { вырезать имя }
   Delete(s, 1, n);
   n := Pos(' ', s);
   otch := Copy(s, 1, n-1); { вырезать отчество }
   Delete(s, 1, n);          { осталась фамилия }
   s := s + ' ' + name[1] + '.' + otch[1] + '.';
   writeln(s);
end.
```

Задания

«3»: Ввести в одну строку фамилию, имя и отчество, разделив их пробелом. Вывести инициалы и фамилию.

Пример:

```
Введите фамилию, имя и отчество:
Иванов Петр Семёнович
Результат:
П.С. Иванов
```

«4»: Ввести имя файла (возможно, без расширения) и изменить его расширение на «.exe».

Пример:

```
Введите имя файла:
qqq
Результат:
qqq.exe
```

```
Введите имя файла:
qqq.com
Результат:
qqq.exe
```

Задания

«5»: Ввести путь к файлу и «разобрать» его, выводя каждую вложенную папку с новой строки

Пример:

```
Введите путь к файлу:
C:\Мои документы\10-Б\Вася\qq.exe
Результат:
C:
Мои документы
10-Б
Вася
qq.exe
```

Задачи на обработку строк

Задача: с клавиатуры вводится символьная строка, представляющая собой сумму двух целых чисел, например:

```
12+35
```

Вычислить эту сумму:

```
12+35=47
```

Алгоритм:

1) найти знак «+»
2) выделить числа слева и справа в отдельные строки
3) перевести строки в числа
4) сложить
5) вывести результат

Преобразования «строка»-«число»

Из строки в число:

```
var N, r: integer;
    X: real;
    s: string;
```

```
s := '123';
Val ( s, N, r ); { N = 123 }
 { r = 0, если ошибки не было
   r – номер ошибочного символа}
s := '123.456';
Val ( s, X, r ); { X = 123.456 }
```

Из числа в строку:

```
N := 123;
Str ( N, s );          { '123' }
X := 123.456;
Str ( X, s );          { '1.234560E+002' }
Str ( X:10:3, s ); { '   123.456' }
```

Программа

```
program qq;
var s, s1, s2: string;
    r, n, n1, n2, sum: integer;
begin
  writeln('Введите выражение (сумму чисел):');
  readln(s);
  n:=Pos('+', s);
  s1:=Copy(s, 1, n-1);
  s2:=Copy(s, n+1, Length(s)-n);
  Val(s1, n1, r);
  Val(s2, n2, r);
  sum:=n1+n2;
  writeln(n1, '+', n2, '=', sum);
end.
```

слагаемые-строки

сумма

слагаемые-числа

слагаемые-строки

слагаемые-числа

Задания

«3»: Ввести арифметическое выражение: разность двух чисел. Вычислить эту разность.

Пример:

```
25-12
Ответ: 13
```

«4»: Ввести арифметическое выражение: сумму трёх чисел. Вычислить эту сумму.

Пример:

```
25+12+34
Ответ: 71
```

Задания

«5»: Ввести арифметическое выражение с тремя числами, в котором можно использовать сложение и вычитание. Вычислить это выражение.

Пример:

```
25+12+34
```

```
Ответ: 71
```

Пример:

```
25+12-34
```

```
Ответ: 3
```

Пример:

```
25-12+34
```

```
Ответ: 47
```

Пример:

```
25-12-34
```

```
Ответ: -21
```

Задания

«6»: Ввести арифметическое выражение с тремя числами, в котором можно использовать сложение, вычитание и умножение. Вычислить это выражение.

Пример:

```
25+12*3
```

Ответ: 61

Пример:

```
25*2-34
```

Ответ: 16

Пример:

```
25-12+34
```

Ответ: 47

Пример:

```
25*2*3
```

Ответ: 150

Посимвольный ввод

Задача: с клавиатуры вводится число N, обозначающее количество футболистов команды «Шайба», а затем – N строк, в каждой из которых – информация об одном футболисте таком формате:

<Фамилия> <Имя> <год рождения> <голы>

Все данные разделяются одним пробелом. Нужно подсчитать, сколько футболистов, родившихся в период с 1988 по1990 год, не забили мячей вообще.

Алгоритм:

```
for i:=1 to N do begin
  { пропускаем фамилию и имя }
  { читаем год рождения Year и число голов Gol }
  if (1988 <= Year) and (Year <=1990) and
     (Gol = 0) then  { увеличиваем счетчик }
end;
```

Посимвольный ввод

Пропуск фамилии:

```
var c: char;
```

```
repeat
  read(c);
until c = ' '; { пока не встретим пробел }
```

Пропуск имени:

```
repeat read(c); until c = ' ';
```

Ввод года рождения:

```
var Year: integer;
```

```
read(Year); { из той же введенной строки }
```

Ввод числа голов и переход к следующей строке:

```
readln(Gol); { читать все до конца строки }
```

```
var Gol: integer;
```

Программа

```
program qq;
var c: char;
    i, N, count, Year, Gol: integer;
begin
  writeln('Количество футболистов');
  readln(N);
  count := 0;
  for i:=1 to N do begin
    repeat read(c); until c = ' ';
    repeat read(c); until c = ' ';
    read(Year);
    readln(Gol);
    if (1988 <= Year) and (year <= 1990) and
       (Gol = 0) then count := count + 1;
  end;
  writeln(count);
end.
```

Посимвольный ввод

Если фамилия нужна:

```
var fam: string;
```

```
fam := '';   { пустая строка }
repeat
  read(c);              { прочитать символ }
  fam := fam + c;       { прицепить к фамилии }
until c = ' ';
```

Вместо read(Year):

```
var s: string;
```

```
s := '';   { пустая строка }
repeat
  read(c);            { прочитать символ }
  s := s + c;         { прицепить к году }
until c = ' ';
Val(s, Year, r); { строку – в число }
```

Посимвольный ввод

Если нужно хранить все фамилии:

массив символьных строк

```
const MAX = 100;
var fam: array[1..MAX] of string;
...
fam[i] := '';  { пустая строка }
repeat
  read(c);          { прочитать символ }
  fam[i] := fam[i] + c;
until c = ' ';
```

Задания

Информация о футболистах вводится так же, как и для приведенной задачи (сначала N, потом N строк с данными).

«3»: Вывести фамилии и имена всех футболистов, которые забили больше двух голов.

Пример:

```
Иванов Василий
Семёнов Кузьма
```

«4»: Вывести фамилию и имя футболиста, забившего наибольшее число голов, и количество забитых им голов.

Пример:

```
Иванов Василий 25
```

Задания

«5»: Вывести *в алфавитном порядке* фамилии и имена всех футболистов, которые забили хотя бы один гол. В списке не более 100 футболистов.

Пример:

```
Васильев Иван
Иванов Василий
Кутузов Михаил
Пупкин Василий
```

Программирование на языке Паскаль
Часть II

Тема 7. Рекурсивный перебор

Рекурсивный перебор

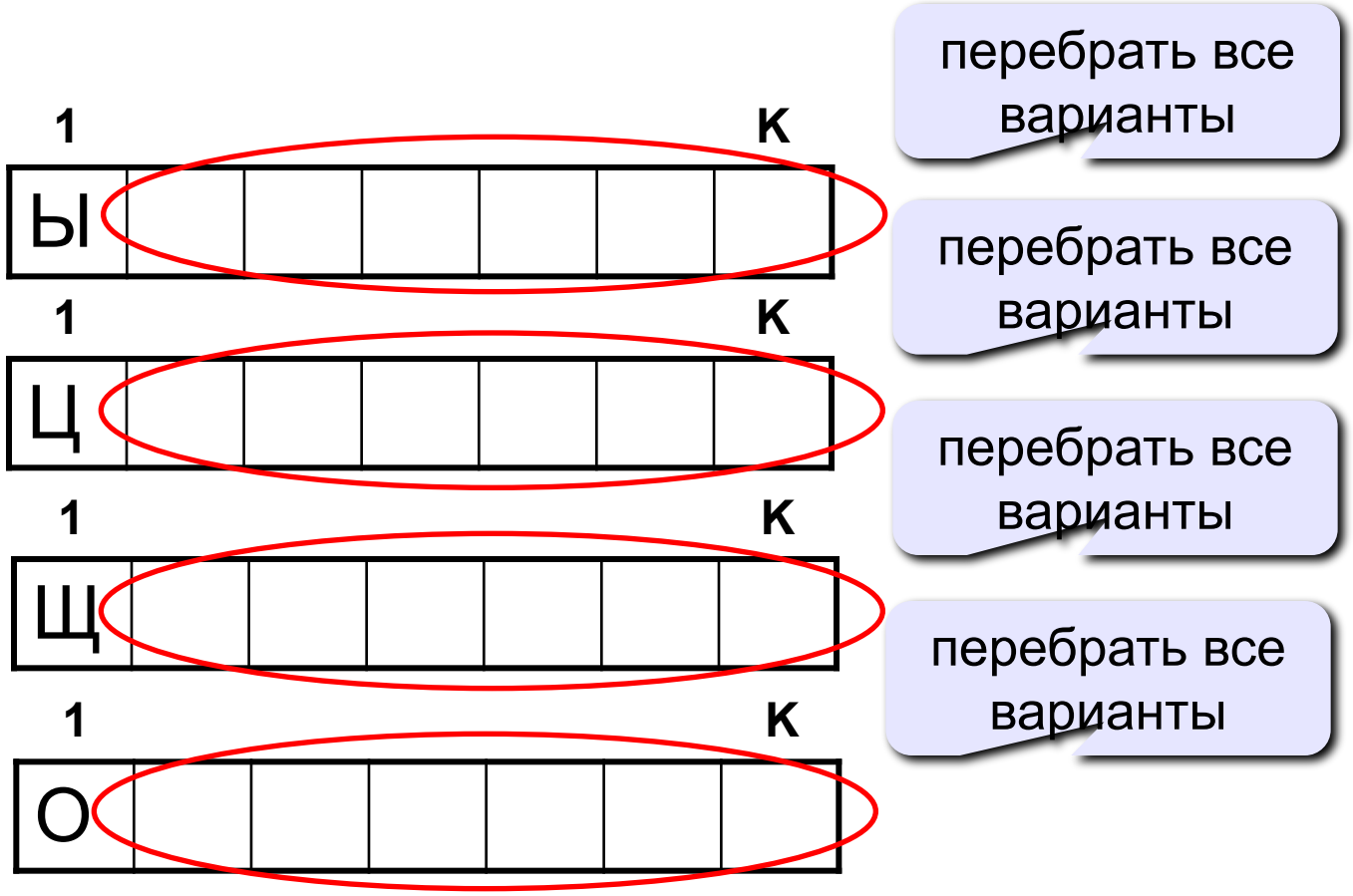
Задача: Алфавит языка племени «тумба-юмба» состоит из букв **Ы**, **Ц**, **Щ** и **О**. Вывести на экран все слова из **K** букв, которые можно составить в этом языке, и подсчитать их количество. Число **K** вводится с клавиатуры.

Количество вариантов:

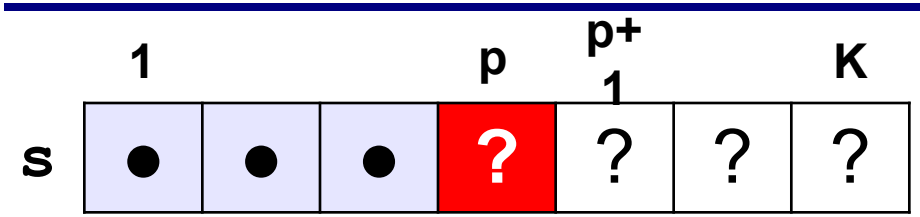
$$N = 4 \cdot 4 \cdot 4 \cdot \square \cdot 4 = 4^K$$

Рекурсивный перебор

Рекурсия: Решения задачи для слов из **K** букв сводится к 4-м задачам для слов из **K-1** букв.



Процедура



Глобальные переменные:

```
var s: string;
    count, K: integer;
```

```
procedure Rec(p: integer);
begin
  if p > K then begin
    writeln(s);
    count := count+1;
  end
  else begin
    s[p]:='Ы'; Rec ( p+1 );
    s[p]:='Ц'; Rec ( p+1 );
    s[p]:='Щ'; Rec ( p+1 );
    s[p]:='О'; Rec ( p+1 );
  end;
end;
```

окончание рекурсии

рекурсивные вызовы

? А если букв много?

Процедура

```
procedure Rec(p: integer);
const letters = 'ЫЩЩО';
var i: integer;
begin
  if p > k then begin
     writeln(s);
     count := count+1;
  end
  else begin
     for i:=1 to length(letters) do begin
       s[p] := letters[i];
       Rec(p+1);
     end;
  end;
end;
```

все буквы

локальная переменная

цикл по всем буквам

Программа

```
program qq;
var s: string;
    K, i, count: integer;
    procedure Rec(p: integer);
    ...
    end;
begin
  writeln('Введите длину слов:');
  read ( K );
  s := '';
  for i:=1 to K do s := s + ' ';
  Rec ( 1 );
  writeln('Всего ', count, ' слов');
end.
```

глобальные переменные

процедура

строка из K пробелов

Задания

Алфавит языка племени «тумба-юмба» состоит из букв **Ы**, **Ц**, **Щ** и **О**. Число **К** вводится с клавиатуры.

«3»: Вывести на экран все слова из **К** букв, в которых первая буква – **Ы**, и подсчитать их количество.

«4»: Вывести на экран все слова из **К** букв, в которых буква **Ы** встречается более 1 раза, и подсчитать их количество.

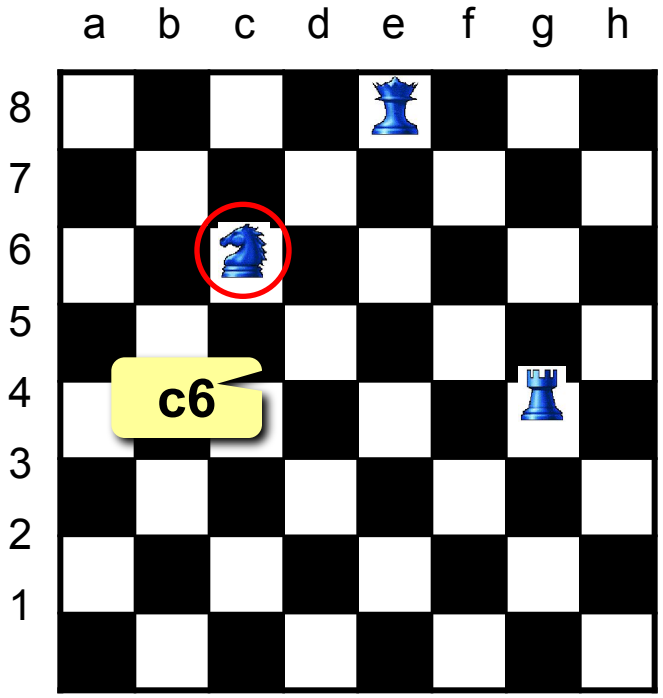
«5»: Вывести на экран все слова из **К** букв, в которых есть одинаковые буквы, стоящие рядом (например, **ЫЩЩО**), и подсчитать их количество.

Программирование на языке Паскаль
Часть II

Тема 8. Матрицы

Матрицы

Задача: запомнить положение фигур на шахматной доске.



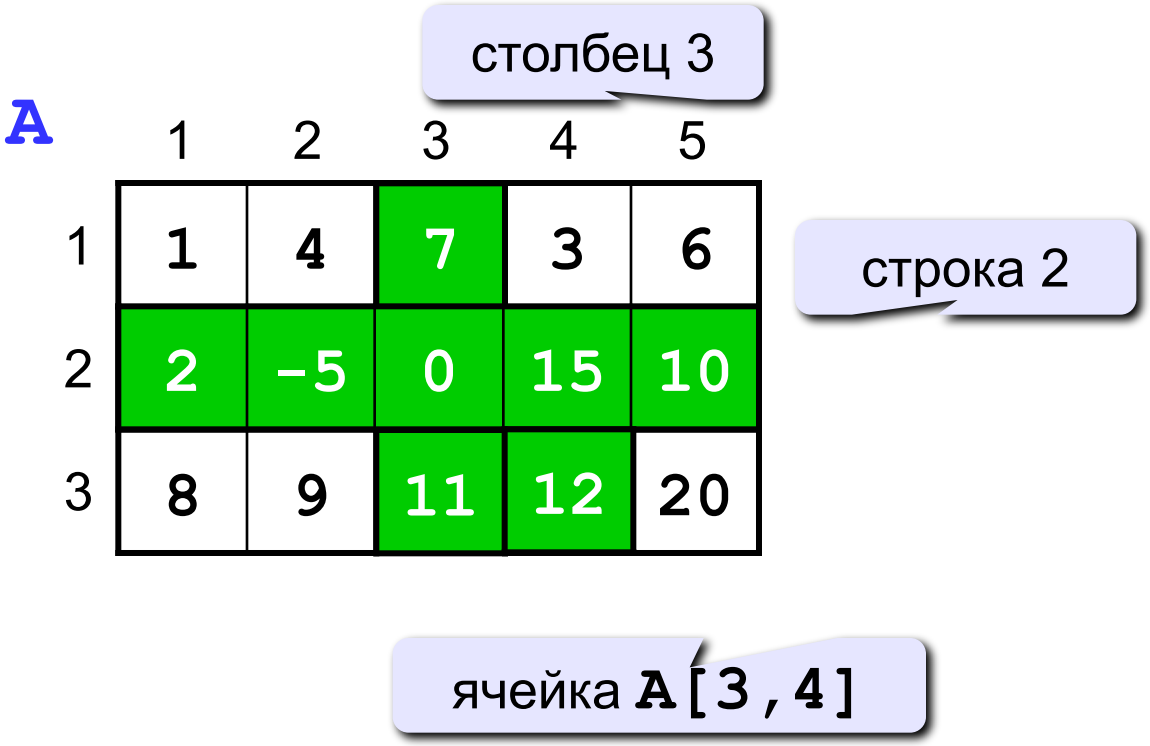
	1	2	3	4	5	6	7	8
8	0	0	0	0	2	0	0	0
7	0	0	0	0	0	0	0	0
6	0	0	3	0	0	0	0	0
5	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	4	0
3	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0

A[6,3]

Матрицы

Матрица – это прямоугольная таблица чисел (или других элементов одного типа).

Матрица – это массив, в котором каждый элемент имеет два индекса (номер строки и номер столбца).



Матрицы

Объявление:

```
const N = 3;
      M = 4;
var A: array[1..N,1..M] of integer;
    B: array[-3..0,-8..M] of integer;
    Q: array['a'..'d',False..True] of real;
```

Ввод с клавиатуры:

Если переставить циклы?

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

i j

A[1,1] 2
A[1,2] 5
A[1,3] 1
= 4
...
A[3,4] 5
= 4

Матрицы

Заполнение случайными числами

```
for i:=1 to N do
  for j:=1 to M do
    A[i,j] := random(25) - 10;
```

цикл по строкам

цикл по столбцам

тервал?

Вывод на экран

```
for i:=1 to N do begin
  for j:=1 to M do
    write ( A[i,j]:5 );
  writeln;
end;
```

вывод строки

```
   12   25    1   13
  156    1   12  447
    1  456  222   23
```

в той же строке

перейти на новую строку

Если переставить циклы?

Обработка всех элементов матрицы

Задача: заполнить матрицу из 3 строк и 4 столбцов случайными числами и вывести ее на экран. Найти сумму элементов матрицы.

```
program qq;
const N = 3; M = 4;
var A: array[1..N,1..M] of integer;
    i, j, S: integer;
begin
  { заполнение матрицы и вывод на экран}
  S := 0;
  for i:=1 to N do
    for j:=1 to M do
      S := S + A[i,j];
  writeln('Сумма элементов матрицы ', S);
end.
```

Задания

Заполнить матрицу из 8 строк и 5 столбцов случайными числами в интервале [-10,10] и вывести ее на экран.

«3»: Удвоить все элементы матрицы и вывести её на экран.

«4»: Найти минимальный и максимальный элементы в матрице их номера. Формат вывода:

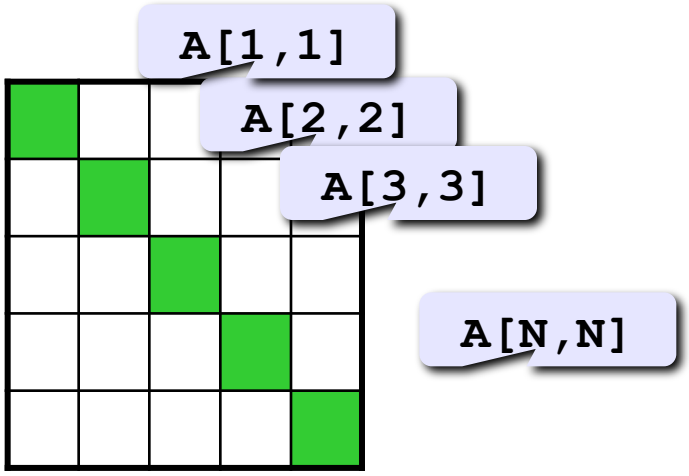
```
Минимальный элемент  A[3,4]=-6
Максимальный элемент A[2,2]=10
```

«5»: Вывести на экран строку, сумма элементов которой максимальна. Формат вывода:

```
Строка 2:  3  5  8  9  8
```

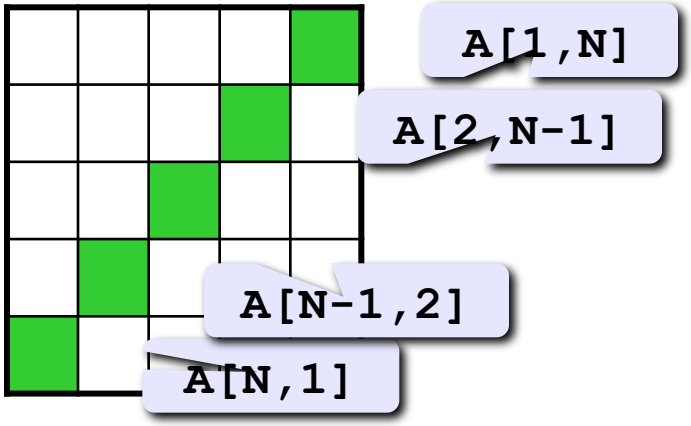
Операции с матрицами

Задача 1. Вывести на экран главную диагональ квадратной матрицы из N строк и N столбцов.




```
for i:=1 to N do
  write ( A[i,i]:5 );
```

Задача 2. Вывести на экран вторую диагональ.

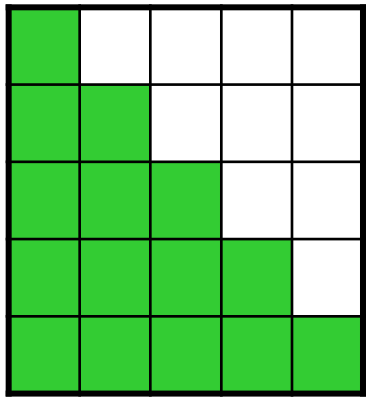


сумма номеров строки и столбца N+1

```
for i:=1 to N do
  write ( A[i, N+1-i ]:5 );
```

Операции с матрицами

Задача 3. Найти сумму элементов, стоящих на главной диагонали и ниже ее.



Одиночный цикл или вложенный?

строка 1: `A[1,1]`

строка 2: `A[2,1]+A[2,2]`

`...`

строка N: `A[N,1]+A[N,2]+...+A[N,N]`

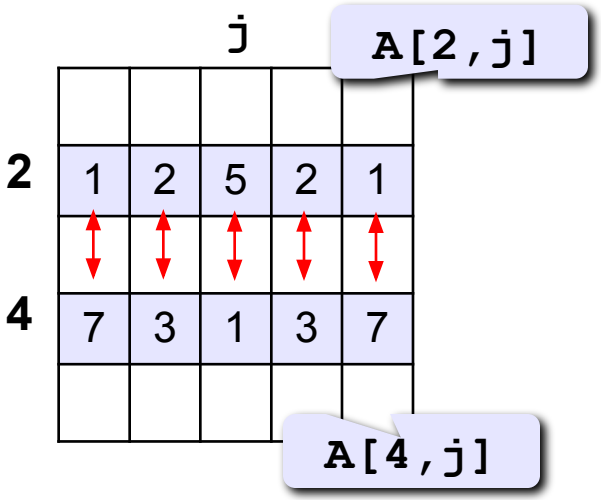
```
S := 0;
for i:=1 to N do
  for j:=1 to i do
    S := S + A[i,j];
```

цикл по всем строкам

складываем нужные элементы строки **i**

Операции с матрицами

Задача 4. Перестановка строк или столбцов. В матрице из N строк и M столбцов переставить 2-ую и 4-ую строки.




```
for j:=1 to M do begin
  c := A[2,j];
  A[2,j] := A[4,j];
  A[4,j] := c;
end;
```

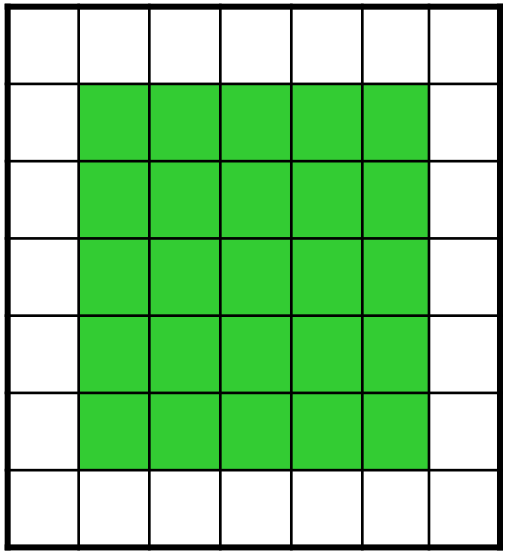
Задача 5. К третьему столбцу добавить шестой.

```
for i:=1 to N do
  A[i,3] := A[i,3] + A[i,6];
```

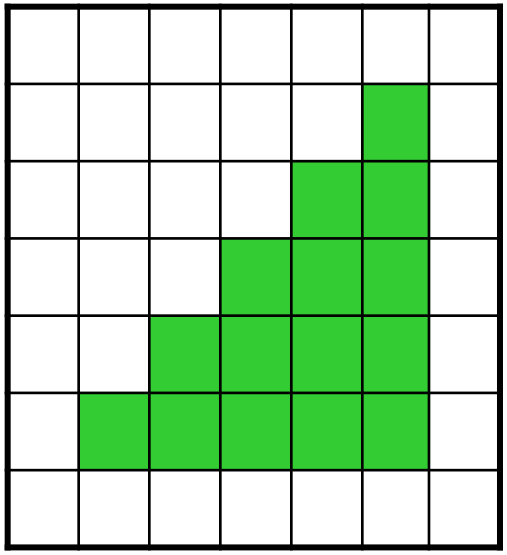
Задания

Заполнить матрицу из 7 строк и 7 столбцов случайными числами в интервале [10,90] и вывести ее на экран. Заполнить элементы, отмеченные зеленым фоном, числами 99, и вывести полученную матрицу на экран.

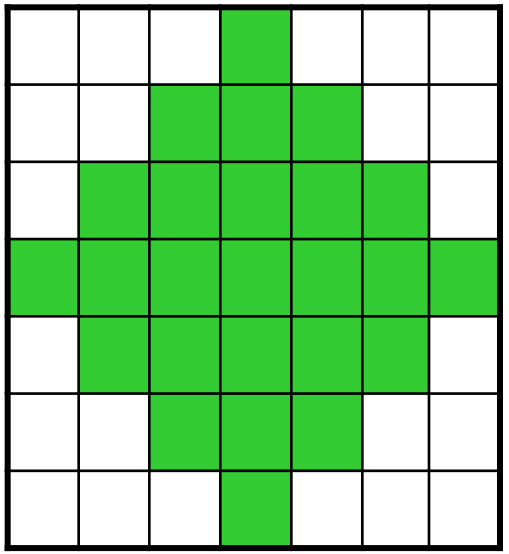
«3»:



«4»:



«5»:

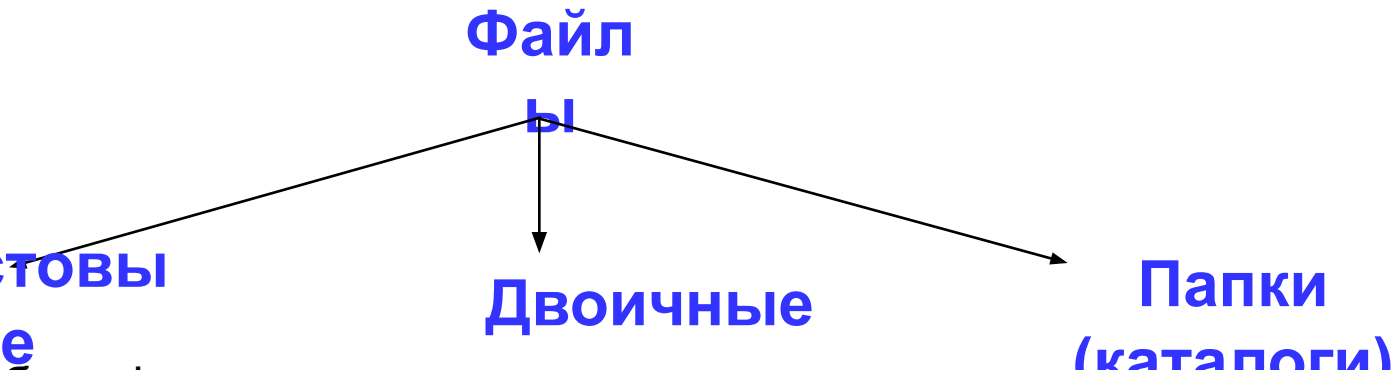


Программирование на языке Паскаль
Часть II

Тема 9. Файлы

Файлы

Файл – это область на диске, имеющая имя.



только текст без оформления, не содержат управляющих символов (с кодами < 32)

ACSII (1 байт на символ)

UNICODE (2 байта на символ)

```
*.txt, *.log,
*.htm, *.html
```

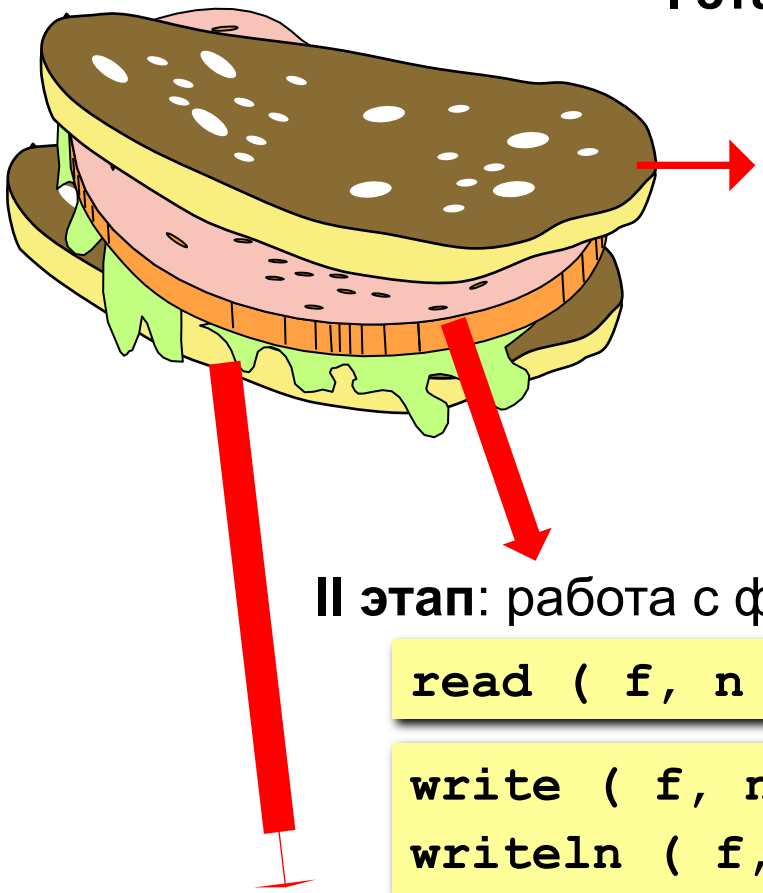
могут содержать любые символы кодовой таблицы

```
*.doc, *.exe,
*.bmp, *.jpg,
*.wav, *.mp3,
*.avi, *.mpg
```

Принцип сэндвича

Переменная типа «текстовый файл»:

```
var f: text;
```



I этап. открыть файл :

- связать переменную **f** с файлом

```
assign(f, 'qq.txt');
```

- открыть файл (сделать его активным, приготовить к работе)

```
reset(f); {для чтения}
```

```
rewrite(f); {для записи}
```

II этап: работа с файлом

```
read ( f, n );  { ввести значение n }
```

```
write ( f, n );  { записать значение n }
writeln ( f, n );{с переходом на нов.строку }
```

III этап: закрыть файл

```
close(f);
```

Работа с файлами

Особенности:

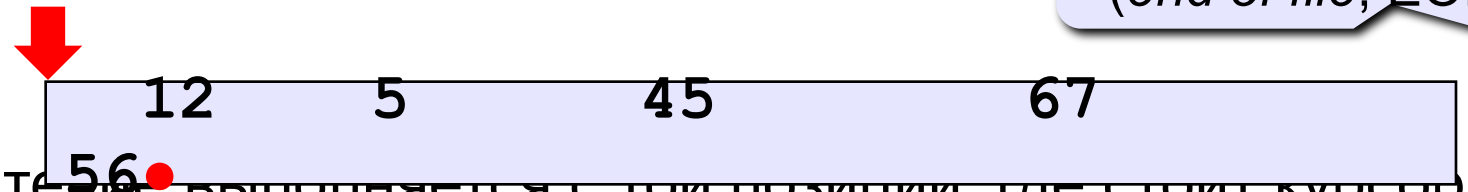
- имя файла упоминается только в команде **`assign`**, обращение к файлу идет через файловую переменную
- файл, который открывается на чтение, должен **существовать**
- если файл, который открывается на запись, существует, старое содержимое **уничтожается**
- данные записываются в файл в текстовом виде
- при завершении программы все файлы закрываются автоматически
- после закрытия файла переменную **`f`** можно использовать еще раз для работы с другим файлом

Последовательный доступ

- при открытии файла курсор устанавливается в начало

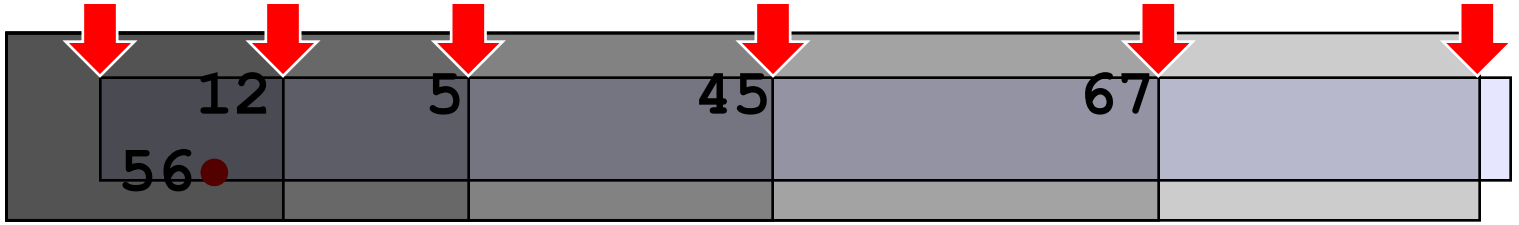
```
assign ( f, 'qq.txt' );
reset ( f );
```

конец файла (*end of file*, EOF)



- чтение выполняется с той позиции, где стоит курсор
- после чтения курсор сдвигается на первый непрочитанный символ

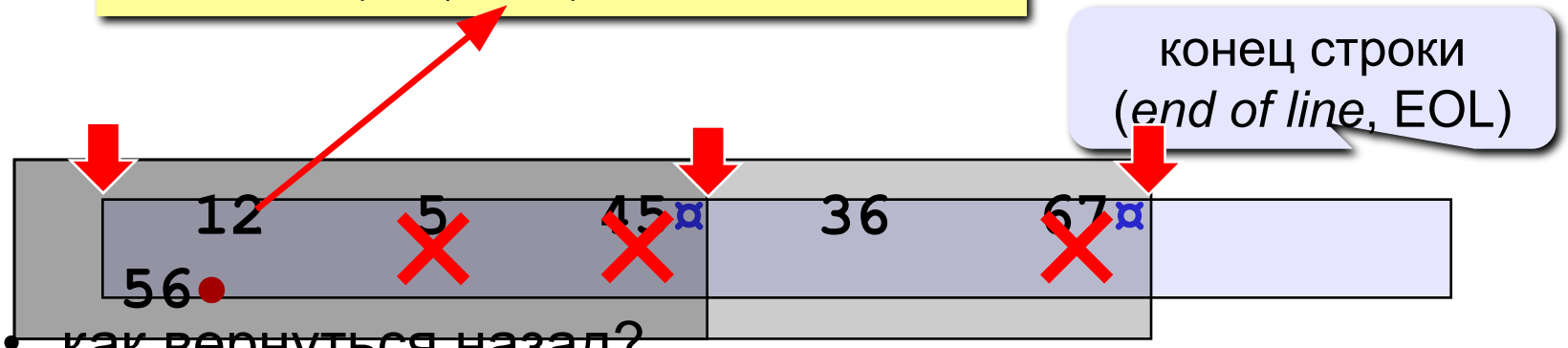
```
read ( f, x );
```



Последовательный доступ

- чтение до конца строки

```
readln ( f, x );
```



- как вернуться назад?

```
close ( f );
reset ( f ); { начать с начала }
```

Пример

Задача: в файле `input.txt` записаны числа (в столбик), сколько их – неизвестно. Записать в файл `output.txt` их сумму.

Можно ли обойтись без массива?

Алгоритм:

1. Открыть файл **`input.txt`** для чтения.
2. **`S := 0;`**
3. Если чисел не осталось, перейти к шагу 7.
4. Прочитать очередное число в переменную **`x`**.
5. **`S := S + x;`**
6. Перейти к шагу 3.
7. Закрыть файл **`input.txt`**.
8. Открыть файл **`output.txt`** для записи.
9. Записать в файл значение **`S`**.
10. Закрыть файл **`output.txt`**.

цикл с условием **«пока есть данные»**

Программа

```
program qq;
var s, x: integer;
    f: text;
begin
   assign(f, 'input.txt');
   reset(f);
   s := 0;
   while not eof(f) do begin
     readln(f, x);
     s := s + x;
   end;
   close(f);
   assign(f, 'output.txt');
   rewrite(f);
   writeln(f, 'Сумма чисел ', s);
   close(f);
end.
```

логическая функция, возвращает **True**, если достигнут конец файла

запись результата в файл **output.txt**

Задания

В файле `data.txt` записаны числа, сколько их – неизвестно.

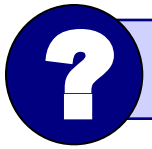
«3»: Найти сумму чётных чисел и записать её в файл **`output.txt`**.

«4»: Найти минимальное и максимальное из четных чисел и записать их в файл **`output.txt`**.

«5»: Найти длину самой длинной цепочки одинаковых чисел, идущих подряд, и записать её в файл **`output.txt`**.

Обработка массивов

Задача: в файле `input.txt` записаны числа (в столбик), сколько их – неизвестно, но не более 100. Переставить их в порядке возрастания и записать в файл `output.txt`.



Можно ли обойтись без массива?

Проблемы:

1. для сортировки надо удерживать в памяти все числа сразу (массив);
2. сколько чисел – неизвестно.

Решение:

3. выделяем в памяти массив из 100 элементов;
4. записываем прочитанные числа в массив и считаем их в переменной `N`;
5. сортируем первые `N` элементов массива;
6. записываем их в файл.

Чтение данных в массив

Глобальные переменные:

```
var A: array[1..100] of integer;
    f: text;
```

Функция: ввод массива, возвращает число элементов

```
function ReadArray: integer;
var i: integer;
begin
  assign(f, 'input.txt');
  reset(f);
  i := 0;
  while (not eof(f)) and (i < 100) do begin
    i := i + 1;
    readln(f, A[i]);
  end;
  close(f);
  ReadArray :=
        i;
end;
```

цикл заканчивается, если достигнут конец файла или прочитали 100 чисел

Программа

```
program qq;
var A: array[1..100] of integer;
    f: text;
    N, i: integer;
function ReadArray: integer;
...
end;
Begi

   N := ReadArray;
   { сортировка первых N элементов }

   assign(f, 'output.txt');
   rewrite(f);
   for i:=1 to N do
     writeln(f, A[i]);
   close(f);
enc
```

вывод отсортированного массива в файл

Задания

В файле `input.txt` записаны числа (в столбик), известно, что их не более 100.

«3»: Отсортировать массив по убыванию и записать его в файл **`output.txt`**.

«4»: Отсортировать массив по убыванию последней цифры и записать его в файл **`output.txt`**.

«5»: Отсортировать массив по возрастанию суммы цифр и записать его в файл **`output.txt`**.

Обработка текстовых данных

Задача: в файле `input.txt` записаны строки, в которых есть слово-паразит «*короче*». Очистить текст от мусора и записать в файл `output.txt`.

Файл `input.txt` :

```
Мама, короче, мыла, короче, раму.
Декан, короче, пропил, короче, бутан.
А роза, короче, упала на лапу, короче, Азора.
Каждый, короче, охотник желает, короче, знать, где ...
```

Результат - файл `output.txt` :

```
Мама мыла раму.
Декан пропил бутан.
А роза упала на лапу Азора.
Каждый охотник желает знать, где сидит фазан.
```

Обработка текстовых данных

Алгоритм:

пока не кончились данные

1. Прочитать строку из файла (`readln`).
2. Удалить все сочетания ", *короче,*" (`Pos, Delete`).
3. Записать строку в другой файл.
4. Перейти к шагу 1.

Обработка строки `s`:

искать «, короче,»

```
repeat
  i := Pos(', короче,', s);
  if i <> 0 then Delete(s, i, 9);
until i = 0;
```

удалить 9 символов

Особенности:

надо одновременно держать открытыми два файла (один в режиме чтения, второй – в режиме записи).

Работа с двумя файлами одновременно

```
program qq;
var s: string;
    i: integer;
    fIn, fOut:
        text;
begin
  assign(fIn, 'input.txt');
  reset(fIn);
  assign(fOut, 'output.txt');
  rewrite(fOut);
  { обработать файл }
  close(fIn);
  close(fOut);
end.
```

файловые переменные

открыть файл для чтения

открыть файл для записи

Полный цикл обработки файла

пока не достигнут конец файла

```
while not eof(fIn) do begin
  readln(fIn, s);
  repeat
    i := Pos(', короче,', s);
    if i <> 0 then
      Delete(s, i, 9);
  until i = 0;
end;
```

обработка строки

запись «очищенной» строки

Задания

В файле `input.txt` записаны строки, сколько их – неизвестно.

«3»: Заменить все слова «короче» на «в общем» и записать результат в файл **`output.txt`**.

«4»: Вывести в файл **`output.txt`** только те строки, в которых есть слово «пароход». В этих строках заменить все слова «короче» на «в общем».

«5»: Вывести в файл **`output.txt`** только те строки, в которых больше 5 слов (слова могут быть разделены несколькими пробелами).

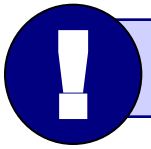
Сортировка списков

Задача: в файле `list.txt` записаны фамилии и имена пользователей сайта (не более 100). Вывести их в алфавитном порядке в файл `sort.txt`.

Файл `list.txt` :

```
Федоров Иван
Иванов Федор
Анисимов Никита
Никитин Николай
```

Нужен ли массив!



Для сортировки нужен массив!

Результат – файл `sort.txt` :

```
Анисимов Никита
Иванов Федор
Никитин Николай
Федоров Иван
```

Сортировка списков

Алгоритм:

1)прочитать строки из файла в массив строк, подсчитать их в переменной **N**

2)отсортировать первые **N** строк массива по алфавиту

3)вывести первые **N** строк массива в файл

Объявление массива (с запасом):

```
const MAX = 100;
var s: array[1..MAX] of string;
```

Сортировка списков

Ввод массива строк из файла:

```
var f:Text;
    N: integer;
```

```
assign(f, 'list.txt');
reset(f);
N:=0;
while not eof(f) do begin
  N:=N+1;
  readln(f, s[N]);
end;
close(f);
```

Сортировка списков

Сортировка первых N элементов массива:

```
for i:=1 to N-1 do begin
  nMin:=i;
  for j:=i+1 to N do
    if s[j]<s[nMin] then nMin:=j;
  if i<>nMin then begin
    c:=s[i];
    s[i]:=s[nMin];
    s[nMin]:=c;
  end;
end;
```

```
var i, j, nMin:
    integer;
  c: string;
```

Какой метод?

Сортировка списков

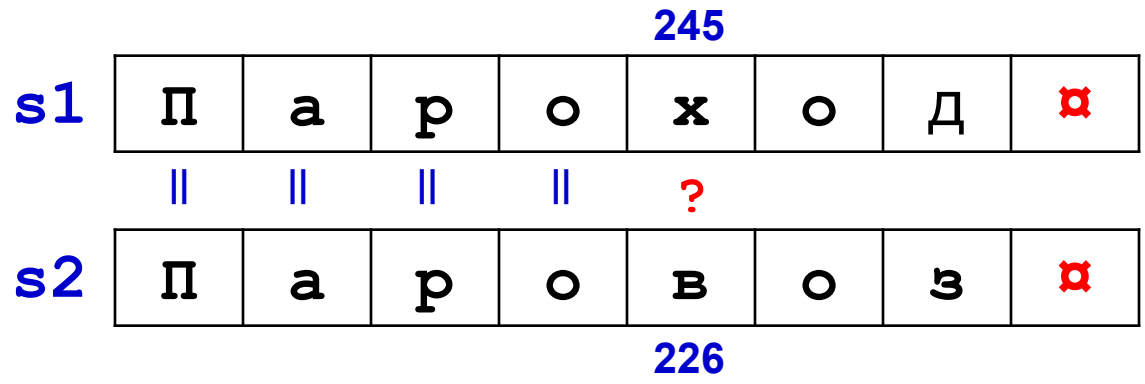
Вывод первых N строк массива в файл:

```
assign(f, 'sort.txt');
rewrite(f);
for i:=1 to N do
  writeln(f, s[i]);
close(f);
```

```
var f:Text;
    i, N: integer;
```

Сортировка списков

Как сравниваются строки:



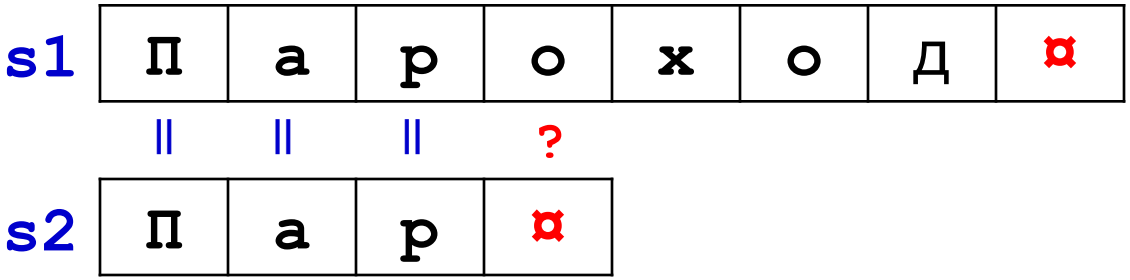
Кодовая таблица:

	А	Б	В	…	Я	а	б	в	…	х	…	я
Win	192	193	194	…	223	224	225	226	…	245	…	255
UNICODE	1040	1041	1042	…	1071	1072	1073	1074	…	1093	…	1103

```
код('х') > код('в')
'х' > 'в'
'Пароход' > 'Паровоз'
```

Сортировка списков

Как сравниваются строки:




```
'x' > ¤

'Пароход' > 'Пар'
```

Сортировка списков

Работа с отдельной строкой массива:

```
var s: array[1..MAX] of string;
    c: string; {вспомогательная строка}
...
for i:=1 to N do begin
  c:= s[i];
    { работаем со строкой c, меняем ее }
  s[i]:= c;
end;
```

Задания

«3»: Добавить к списку нумерацию:

```
1) Анисимов Никита
2) Иванов Федор
```

«4»: Выполнить задачу на «3» и сократить имя до первой буквы:

```
1) Анисимов Н.
2) Иванов Ф.
```

«5»: Выполнить задачу на «4», но при выводе начинать с имени:

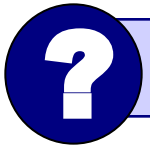
```
1) Н. Анисимов
2) Ф. Иванов
```

Списки с числовыми данными

Задача: в файле `marks.txt` записаны фамилии и имена школьников и баллы, полученные ими на экзамене (0-100). В файле не более 100 строк. Вывести в файл `best.txt` список тех, кто получил более 75 баллов.

Файл `marks.txt`:

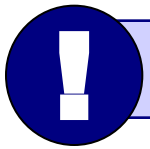
```
Федоров Иван 78
Иванов Федор 63
Анисимов Никита 90
Никитин Николай 55
```



Нужен ли массив!

Результат – файл `best.txt`:

```
Федоров Иван 78
Анисимов Никита 90
```



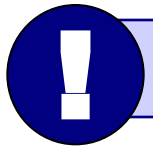
Используем два файла одновременно!

Работа с двумя файлами одновременно

```
var fIn, fOut: Text;
...
assign(fIn, 'marks.txt');
reset(fIn);
assign(fOut, 'best.txt');
rewrite(fOut);
while not eof(fIn) do begin
    { обработка строк из файла }
end;
close(fIn);
close(fOut);
```

Цикл обработки файла

```
var ball: integer;
...
while not eof(fIn) do begin
  readln(fIn, s);
    { обработка строки s }
    { ball:=результат на экзамене }
  if ball > 75 then
    writeln(fOut, s);
end;
```



Оба файла открыты одновременно!

Преобразования «строка»-«число»

Из строки в число:

```
var N, r: integer;
    X: real;
    s: string;
```

```
s := '123';
Val ( s, N, r ); { N = 123 }
 { r = 0, если ошибки не было
   r – номер ошибочного символа}
s := '123.456';
Val ( s, X, r ); { X = 123.456 }
```

Из числа в строку:

```
N := 123;
Str ( N, s );       { '123' }
X := 123.456;
Str ( X, s );       { '1.234560E+002' }
Str ( X:10:3, s ); { '   123.456' }
```

Обработка строки

```
var n, r: integer;
    s, fam, name: string;
```

s:

8	2

```
n:=Pos(' ', s);          { n:=7; }
fam:=Copy(s,1,n-1);      { fam:='Пупкин'; }
Delete(s, 1, n);         { s:='Вася 82'; }
n:=Pos(' ', s);          { n:=5; }
name:=Copy(s,1,n-1);     { name:='Вася'; }
Delete(s, 1, n);           { s:='82'; }
Val(s, ball, r);         { ball:=82; }
```

Задания

«3»: Добавить к списку нумерацию:

```
1) Федоров Иван 78
2) Анисимов Никита 90
```

«4»: Выполнить задачу на «3» и сократить имя до первой буквы:

```
1) Федоров И. 78
2) Анисимов Н. 90
```

«5»: Выполнить задачу на «4», но отсортировать список по алфавиту.

```
1) Анисимов Н. 90
2) Федоров И. 78
```

«6»: Выполнить задачу на «4», но отсортировать список по убыванию отметки (балла).

Конец фильма

ПОЛЯКОВ Константин Юрьевич
д.т.н., учитель информатики высшей категории,
ГОУ СОШ № 163, г. Санкт-Петербург
kpolyakov@mail.ru