

# Программирование на алгоритмическом языке

§ 62. Массивы

§ 63. Алгоритмы обработки массивов

§ 64. Сортировка

§ 65. Двоичный поиск

§ 66. Символьные строки

§ 67. Матрицы

§ 68. Работа с файлами

# Программирование на алгоритмическом языке

## § 62. 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. Каждая ячейка в массиве имеет уникальный номер.

**Надо:**

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

# Выделение памяти (объявление)



Массив = таблица!

МИНИМАЛЬНЫЙ  
ИНДЕКС

**целтаб** A [1 : 5]  
**вещтаб** V [0 : 5]  
**логтаб** L [-5 : 5]  
**симтаб** S [65 : 90]

МАКСИМАЛЬНЫЙ  
ИНДЕКС

размер через  
константу

**цел** N = 10  
**целтаб** A [1 : N]



Зачем?

# Что неправильно?

---

```
целтаб А [1:1  
          0]
```

...

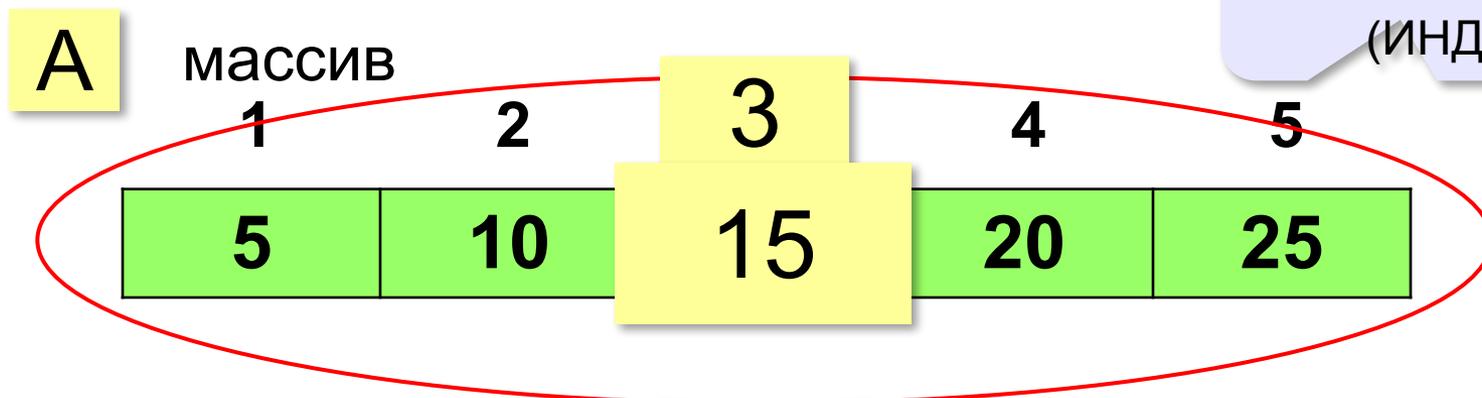
```
А[5] := 4.5;
```

```
целтаб А[1:10]
```

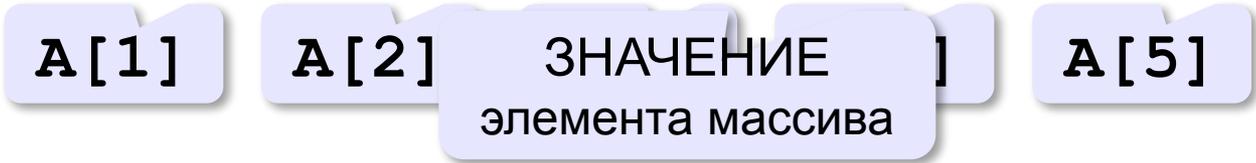
...

```
А[15] := 'a'
```

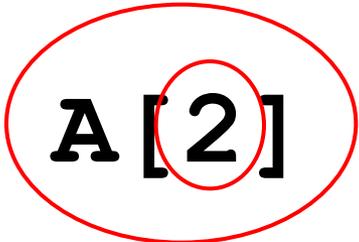
# Обращение к элементу массива



НОМЕР  
элемента массива  
(ИНДЕКС)



НОМЕР (ИНДЕКС)  
элемента массива: 2



ЗНАЧЕНИЕ  
элемента массива: 10

# Как обработать все элементы массива?

Объявление:

цел  $N = 5$

целтаб  $A[1:N]$

Обработка:

| обработать  $A[1]$

| обработать  $A[2]$

| обработать  $A[3]$

| обработать  $A[4]$

| обработать  $A[5]$



1) если  $N$  велико (1000, 1000000)?

2) при изменении  $N$  программа не должна меняться!

# Как обработать все элементы массива?

Обработка с переменной:

```
i := 1
| обработать A[i]
i := i + 1
| обработать A[i]
```

```
i := i + 1
```



Обработка в цикле:

```
i := 1
нц пока i <= N
    | обработать A[i]
    i := i + 1
кц
```

Цикл с переменной:

```
нц для i от 1 до N
    | обработать A[i]
кц
```



# Заполнение массива

алг **Массив**

нач

**цел**  $i$ ,  $N = 10$

**целтаб**  $A[1:N]$

нц для  $i$  от  $1$  до  $N$

$A[i] := i * i$

кц

кон



Чему равен  $A[9]$ ?

# Ввод с клавиатуры и вывод на экран

## Объявление:

```
цел N = 5, i  
целтаб A[1:N]
```

## Ввод с клавиатуры:

```
нц для i от 1 до N  
    вывод 'A[' , i, ']='  
    ввод A[i]  
кц
```

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

## Вывод на экран:

```
вывод 'Массив A', нс  
нц для i от 1 до N  
    вывод A[i], ' '  
кц
```



Зачем пробел?

# Заполнение случайными числами

---

*Задача.* Заполнить массив (псевдо)случайными целыми числами в диапазоне от 20 до 100.

```
нц для i от 1 до N
  A[i] := irand(20, 100)
  вывод A[i], ' '
кц
```

# Перебор элементов

---

## Общая схема:

```
нц для i от 1 до N
    ... | сделать что-то с A[i]
кц
```

## Подсчёт нужных элементов:

**Задача.** В массиве записаны данные о росте баскетболистов. Сколько из них имеет рост больше 180 см, но меньше 190 см?

```
цел count = 0
нц для i от 1 до N
    если 180 < A[i] и A[i] < 190 то
        count := count + 1
    все
кц
```

# Перебор элементов

## Среднее арифметическое:

```
цел count = 0, sum = 0
нц для i от 1 до N
    если 180 < A[i] и A[i] < 190 то
        count := count + 1
        sum := sum + A[i]
    все
кц
вывод sum/count
```

среднее  
арифметическое

# Задачи

---

**«А»:** Заполните массив случайными числами в интервале  $[0,100]$  и найдите среднее арифметическое его значений.

**Пример:**

**Массив :**

**1 2 3 4 5**

**Среднее арифметическое 3.000**

**«В»:** Заполните массив случайными числами в интервале  $[0,100]$  и подсчитайте отдельно среднее значение всех элементов, которые  $<50$ , и среднее значение всех элементов, которые  $\geq 50$ .

**Пример:**

**Массив :**

**3 2 52 4 60**

**Ср. арифм. элементов  $[0, 50)$  : 3.000**

**Ср. арифм. элементов  $[50, 100]$  : 56.000**

# Задачи

---

**«С»:** Заполните массив из  $N$  элементов случайными числами в интервале  $[1, N]$  так, чтобы в массив обязательно вошли все числа от 1 до  $N$  (постройте случайную перестановку).

**Пример:**

**Массив :**

3 2 1 4 5

# Программирование на алгоритмическом языке

## **§ 63. Алгоритмы обработки массивов**

# Поиск в массиве

Найти элемент, равный X:

```
i := 1
нц пока A[i] <> X
  i := i + 1
кц
вывод 'A[' , i , ']=', X
```

 Что плохо?

```
i := 1
нц пока i <= N и A[i] <> X
  i := i + 1
кц
если i <= N то
  вывод 'A[' , i , ']=', X
иначе вывод 'Не нашли!'
все
```

должно быть первым!

 Что если такого нет?

# Поиск в массиве

## Вариант с досрочным выходом:

```
nX := 0
нц для i от 1 до N
  если A[i] = X то
    nX := i
    выход
  все
кц
если nX > 0 то
  вывод 'A[', i, ']=', X
иначе вывод 'Не нашли!'
все
```

досрочный  
выход из  
цикла

# Задачи

---

**«А»:** Заполните массив случайными числами в интервале  $[0,5]$ . Введите число  $X$  и найдите все значения, равные  $X$ .

**Пример:**

Массив :

1 2 3 1 2

Что ищем:

2

Нашли:  $A[2]=2$ ,  $A[5]=2$

**Пример:**

Массив :

1 2 3 1 2

Что ищем:

6

Ничего не нашли.

# Задачи

---

**«В»:** Заполните массив случайными числами в интервале  $[0,5]$ . Определить, есть ли в нем элементы с одинаковыми значениями, стоящие рядом.

**Пример:**

**Массив :**

1 2 3 3 2 1

**Есть : 3**

**Пример:**

**Массив :**

1 2 3 4 2 1

**Нет**

# Задачи

---

**«С»:** Заполните массив случайными числами. Определить, есть ли в нем элементы с одинаковыми значениями, не обязательно стоящие рядом.

**Пример:**

**Массив :**

3 2 1 3 2 5

**Есть :** 3, 2

**Пример:**

**Массив :**

3 2 1 4 0 5

**Нет**

# Максимальный элемент

```
М := А[1]
нц для i от 2 до N
  если А[i] > М то
    М := А[i]
  все
кц
вывод М
```



Как найти его номер?



Что можно улучшить?

```
М := А[1]; nMax := 1
нц для i от 2 до N
  если А[i] > М то
    М := А[i]
    nMax := i
  все
кц
вывод 'А[', nMax, ']=', М
```

# Максимальный элемент и его номер



По номеру элемента можно найти значение!

```
nMax := 1
нц для i от 2 до N
  если A[i] > A[nMax] то
    nMax := i
  все
кц
вывод 'A[', nMax, ']=' , A[nMax]
```

# Задачи

---

**«А»:** Заполнить массив случайными числами и найти минимальный и максимальный элементы массива и их номера.

**Пример:**

**Массив :**

1 2 3 4 5

**Минимальный элемент:  $A[1]=1$**

**Максимальный элемент:  $A[5]=5$**

**«В»:** Заполнить массив случайными числами и найти два максимальных элемента массива и их номера.

**Пример:**

**Массив :**

5 5 3 4 1

**Максимальный элемент:  $A[1]=5$**

**Второй максимум:  $A[2]=5$**

# Задачи

---

**«С»:** Введите массив с клавиатуры и найдите (за один проход) количество элементов, имеющих максимальное значение.

**Пример:**

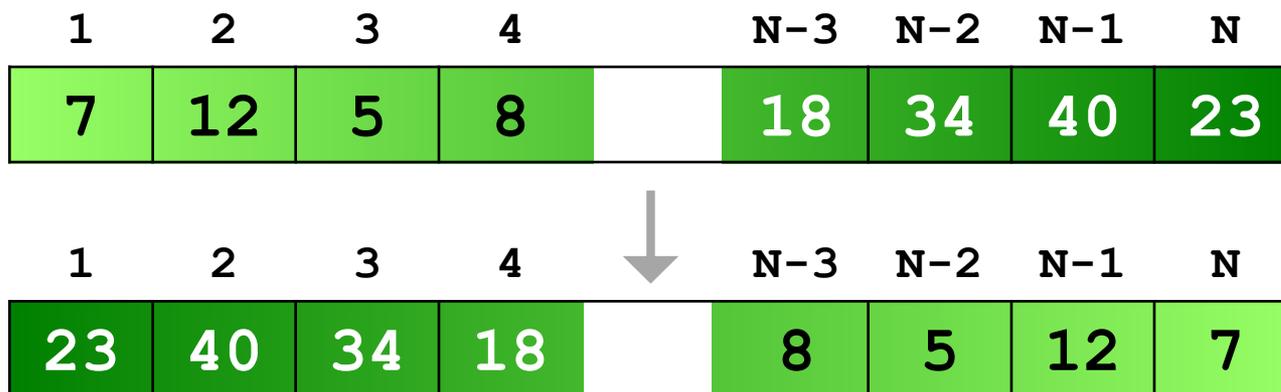
**Массив :**

**3 4 5 5 3 4 5**

**Максимальное значение 5**

**Количество элементов 3**

# Реверс массива



«Простое» решение:

остановиться на середине!

```

нц для i от 1 до div(N, 2)
    | поменять местами A[i] и A[N+1-i]
кц
  
```



Что плохо?

# Реверс массива

---

```
нц для i от 1 до div(N, 2)
  c := A[i]
  A[i] := A[N+1-i]
  A[N+1-i] := c
кц
```



\*Как обойтись без переменной c?

# Циклический сдвиг элементов

1	2	3	4		N-3	N-2	N-1	N
7	12	5	8		18	34	40	23

↓

1	2	3	4		N-3	N-2	N-1	N
12	5	8	15		34	40	23	7

«Простое» решение:

```

нц для i от 1 до N-1
  A[i] := A[i+1]
кц

```

?

Почему не до N?

?

Что плохо?

# Задачи

---

**«А»:** Заполнить массив случайными числами и выполнить циклический сдвиг элементов массива вправо на 1 элемент.

**Пример:**

**Массив :**

1 2 3 4 5 6

**Результат:**

6 1 2 3 4 5

**«В»:** Массив имеет четное число элементов. Заполнить массив случайными числами и выполнить реверс отдельно в первой половине и второй половине.

**Пример:**

**Массив :**

1 2 3 4 5 6

**Результат:**

3 2 1 6 5 4

# Задачи

---

**«С»:** Заполнить массив случайными числами в интервале  $[-100, 100]$  и переставить элементы так, чтобы все положительные элементы стояли в начала массива, а все отрицательные и нули – в конце. Вычислите количество положительных элементов.

**Пример:**

**Массив :**

20 -90 15 -34 10 0

**Результат:**

20 15 10 -90 -34 0

**Количество положительных элементов : 3**

# Отбор нужных элементов

**Задача.** Отобрать элементы массива **A**, удовлетворяющие некоторому условию, в массив **B**.

**«Простое» решение:**

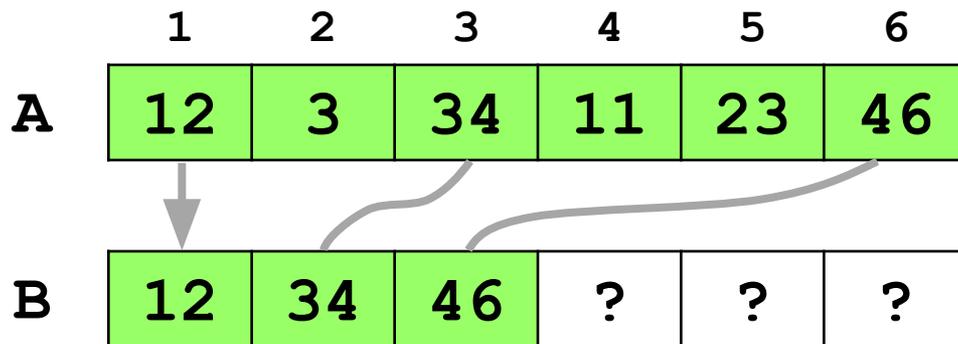
```
нц для  $i$  от 1 до  $N$   
  если условие выполняется для  $A[i]$  то  
     $B[i] := A[i]$   
  все  
кц
```

**?** Что плохо?

	1	2	3	4	5	6
<b>A</b>	12	3	34	11	23	46
	↓		↓		↓	
<b>B</b>	12	?	34	?	?	46

выбрать чётные  
элементы

# Отбор нужных элементов



выбрать чётные  
элементы

```

count := 0 | счётчик
нц для i от 1 до N
  если mod(A[i], 2) = 0 то
    count := count + 1
    В[count] := А[i]
  все
кц
  
```



Если А и В – один и тот же массив?



Как вывести на экран?

```

нц для i от 1 до count
  вывод В[i], ' '
кц
  
```

# Задачи

---

**«А»:** Заполнить массив случайными числами в интервале  $[-10, 10]$  и отобразить в другой массив все чётные отрицательные числа.

**Пример:**

**Массив А:**

-5 6 7 -4 -6 8 -8

**Массив В:**

-4 -6 -8

**«В»:** Заполнить массив случайными числами в интервале  $[0, 100]$  и отобразить в другой массив все простые числа. Используйте логическую функцию, которая определяет, является ли переданное ей число простым.

**Пример:**

**Массив А:**

12 13 85 96 47

**Массив В:**

13 47

# Задачи

---

**«С»:** Заполнить массив случайными числами и отобразить в другой массив все числа Фибоначчи. Используйте логическую функцию, которая определяет, является ли переданное ей число числом Фибоначчи.

**Пример:**

**Массив А:**

12 13 85 34 47

**Массив В:**

13 34

# Программирование на алгоритмическом языке

## § 64. Сортировка

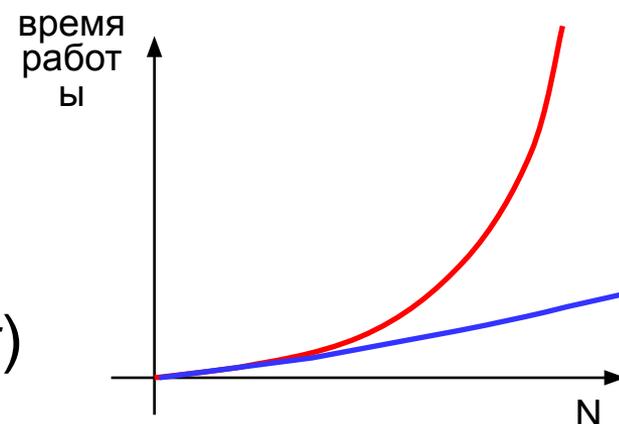
# Что такое сортировка?

**Сортировка** – это расстановка элементов массива в заданном порядке.

...по возрастанию, убыванию, последней цифре, сумме делителей, по алфавиту, ...

## Алгоритмы:

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

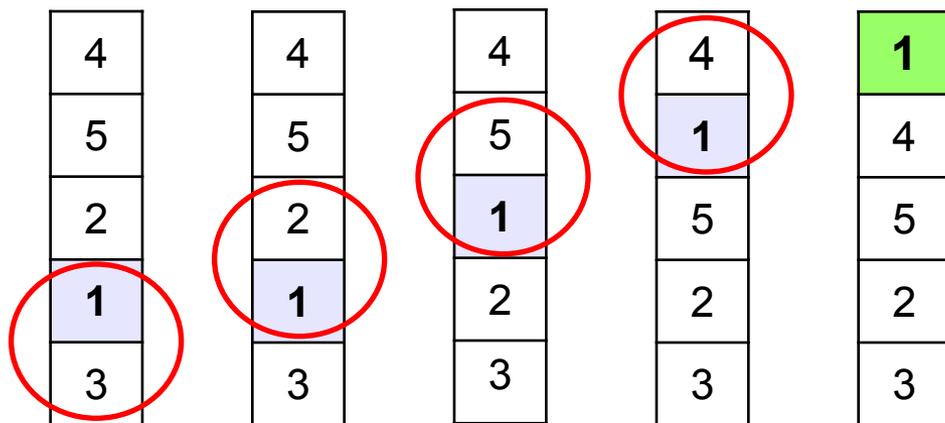


# Метод пузырька (сортировка обменами)

*Идея:* пузырек воздуха в стакане воды поднимается со дна вверх.

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

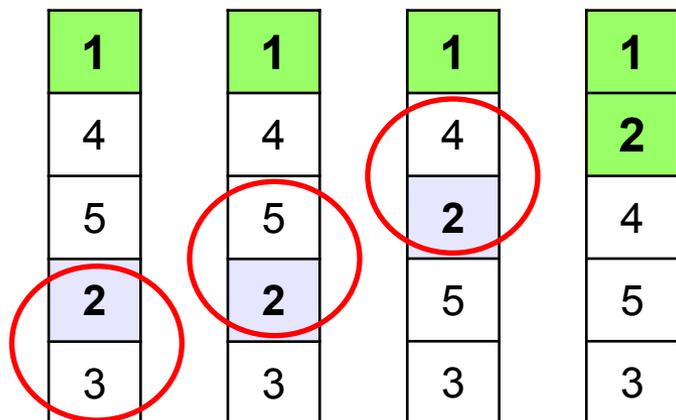
## 1-й проход:



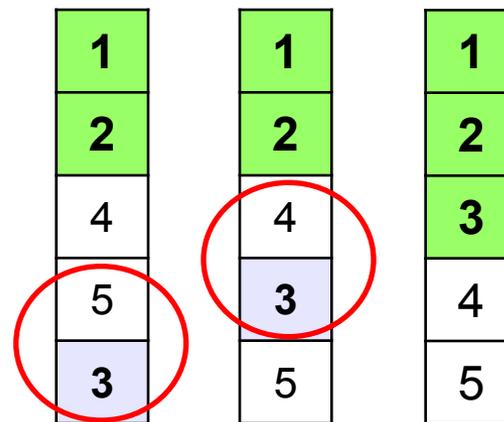
- сравниваем два соседних элемента; если они стоят «неправильно», меняем их местами
- за 1 проход по массиву **один** элемент (самый маленький) становится на свое место

# Метод пузырька

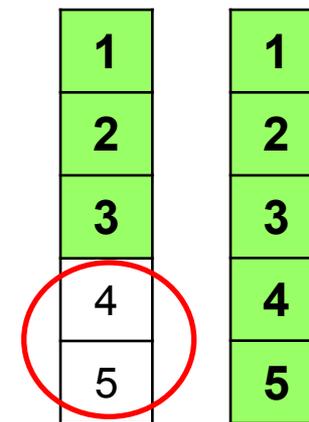
2-й проход:



3-й проход:



4-й проход:



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

# Метод пузырька

## 1-й проход:

```
нц для j от N-1 до 1 шаг -1
  если A[j+1] < A[j] то
    | поменять местами A[j] и A[j+1]
  все
кц
```

единственное  
отличие!

## 2-й проход:

```
нц для j от N-1 до 2 шаг -1
  если A[j+1] < A[j] то
    | поменять местами A[j] и A[j+1]
  все
кц
```

# Метод пузырька

```
нц для i от 1 до N-1
  нц для j от N-1 до i шаг -1
    если A[j+1] < A[j] то
      | поменять местами A[j] и A[j+1]
    все
  кц
кц
```



Как написать метод «камня»?



Как сделать рекурсивный вариант?

# Задачи

---

- «А»: Напишите программу, в которой сортировка выполняется «методом камня» – самый «тяжёлый» элемент опускается в конец массива.
- «В»: Напишите вариант метода пузырька, который заканчивает работу, если на очередном шаге внешнего цикла не было перестановок.
- «С»: Напишите программу, которая сортирует массив по убыванию суммы цифр числа. Используйте функцию, которая определяет сумму цифр числа.

# Метод выбора (минимального элемента)

---

Идея: найти минимальный элемент и поставить его на первое место.

```
нц для i от 1 до N-1
  | найти номер nMin минимального элемента
  | из A[i]..A[N]
  если i <> nMin то
    | поменять местами A[i] и A[nMin]
  все
кц
```

# Метод выбора (минимального элемента)

```
нц для i от 1 до N-1
```

```
  nMin := i
```

```
  нц для j от i+1 до N
```

```
    если A[j] < A[nMin] то
```

```
      nMin := j
```

```
    все
```

```
  кц
```

```
  если i <> nMin то
```

```
    | поменять местами A[i] и A[nMin]
```

```
  все
```

```
кц
```



Как поменять местами два значения?

# Задачи

---

**«А»:** Массив содержит четное количество элементов. Напишите программу, которая сортирует первую половину массива по возрастанию, а вторую – по убыванию. Каждый элемент должен остаться в «своей» половине.

**Пример:**

**Массив :**

5 3 4 2 **1 6 3 2**

**После сортировки :**

2 3 4 5 **6 3 2 1**

# Задачи

---

**«В»:** Напишите программу, которая сортирует массив и находит количество различных чисел в нем.

**Пример:**

**Массив :**

**5 3 4 2 1 6 3 2 4**

**После сортировки:**

**1 2 2 3 3 4 4 5 6**

**Различных чисел: 5**

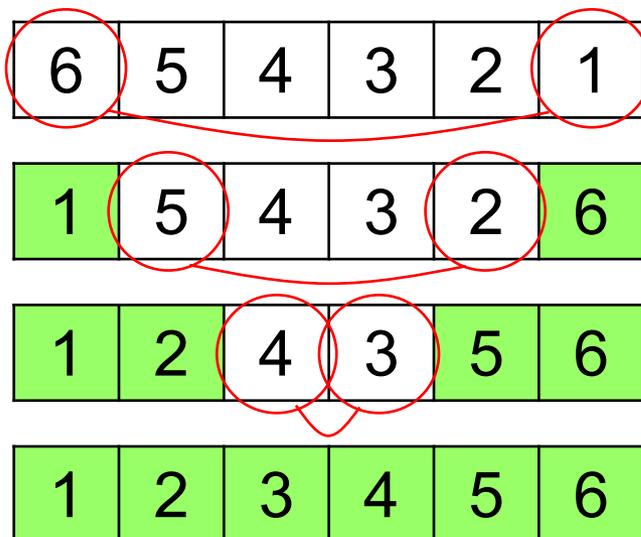
**«С»:** Напишите программу, которая сравнивает число перестановок элементов при использовании сортировки «пузырьком» и методом выбора. Проверьте ее на разных массивах, содержащих 1000 случайных элементов, вычислите среднее число перестановок для каждого метода.

# Быстрая сортировка (*QuickSort*)



Ч.Э.Хоар

*Идея:* выгоднее переставлять элементы, который находятся дальше друг от друга.

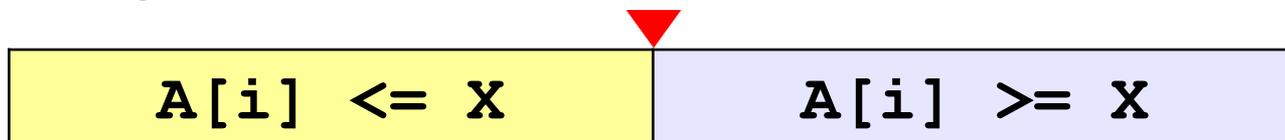


Для массива из  $N$  элементов нужно всего  $N/2$  обменов!

# Быстрая сортировка

**Шаг 1:** выбрать некоторый элемент массива  $X$

**Шаг 2:** переставить элементы так:



при сортировке элементы не покидают « свою область »!

**Шаг 3:** так же отсортировать две получившиеся области

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

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...*).

# Быстрая сортировка

---

## Разделение:

1) выбрать средний элемент массива ( $x=67$ )

78	6	82	67	55	44	34
----	---	----	----	----	----	----

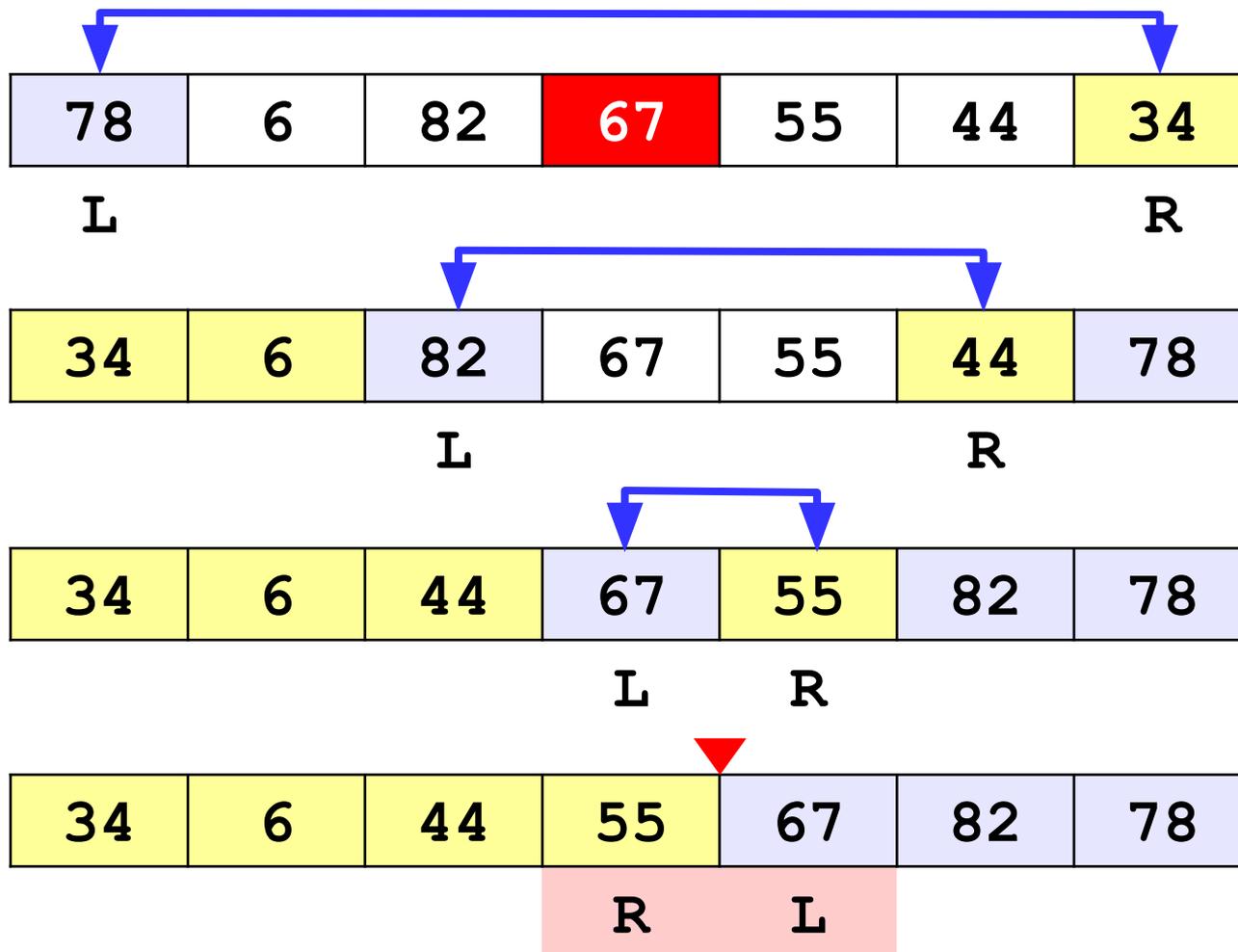
2) установить  $L := 1$ ,  $R := N$

3) увеличивая  $L$ , найти первый элемент  $A[L]$ ,  
который  $\geq x$  (должен стоять справа)

4) уменьшая  $R$ , найти первый элемент  $A[R]$ ,  
который  $\leq x$  (должен стоять слева)

5) если  $L \leq R$  то поменять местами  $A[L]$  и  $A[R]$   
и перейти к п. 3  
иначе **СТОП**.

# Быстрая сортировка



**L > R : разделение закончено!**

# Быстрая сортировка

## Основная программа:

```
цел N = 7  
целтаб A[1:N] }  
алг Быстрая сортировка  
нач  
    | заполнить массив  
    qSort(1, N) | сортировка  
    | вывести результат  
кон
```

глобальные  
данные

# Быстрая сортировка

```
алг qSort(цел nStart, nEnd)
нач
  цел L, R, с, X
  если nStart >= nEnd то выход все
  L := nStart; R := nEnd
  X := A[div(L+R, 2)] | или X := A[irand(L, R)]
  нц пока L <= R | разделение
    нц пока A[L] < X; L := L + 1 кц
    нц пока A[R] > X; R := R - 1 кц
    если L <= R то
      с := A[L]; A[L] := A[R]; A[R] := с
      L := L + 1; R := R - 1
    все
  кц
  qSort(nStart, R) | рекурсивные вызовы
  qSort(L, nEnd)
кон
```

# Быстрая сортировка

Сортировка массива случайных значений:

N	метод пузырька	метод выбора	быстрая сортировка
1000	0,24 с	0,12 с	<b>0,004 с</b>
5000	5,3 с	2,9 с	<b>0,024 с</b>
15000	45 с	34 с	<b>0,068 с</b>

# Задачи

---

**«А»:** Массив содержит четное количество элементов.

Напишите программу, которая сортирует по возрастанию отдельно элементы первой и второй половин массива.

Каждый элемент должен остаться в «своей» половине.

Используйте алгоритм быстрой сортировки.

**Пример:**

**Массив :**

5 3 4 2 1 6 3 2

**После сортировки :**

2 3 4 5 6 3 2 1

# Задачи

---

**«В»:** Напишите программу, которая сортирует массив и находит количество различных чисел в нем. Используйте алгоритм быстрой сортировки.

**Пример:**

**Массив :**

**5 3 4 2 1 6 3 2 4**

**После сортировки:**

**1 2 2 3 3 4 4 5 6**

**Различных чисел: 5**

# Задачи

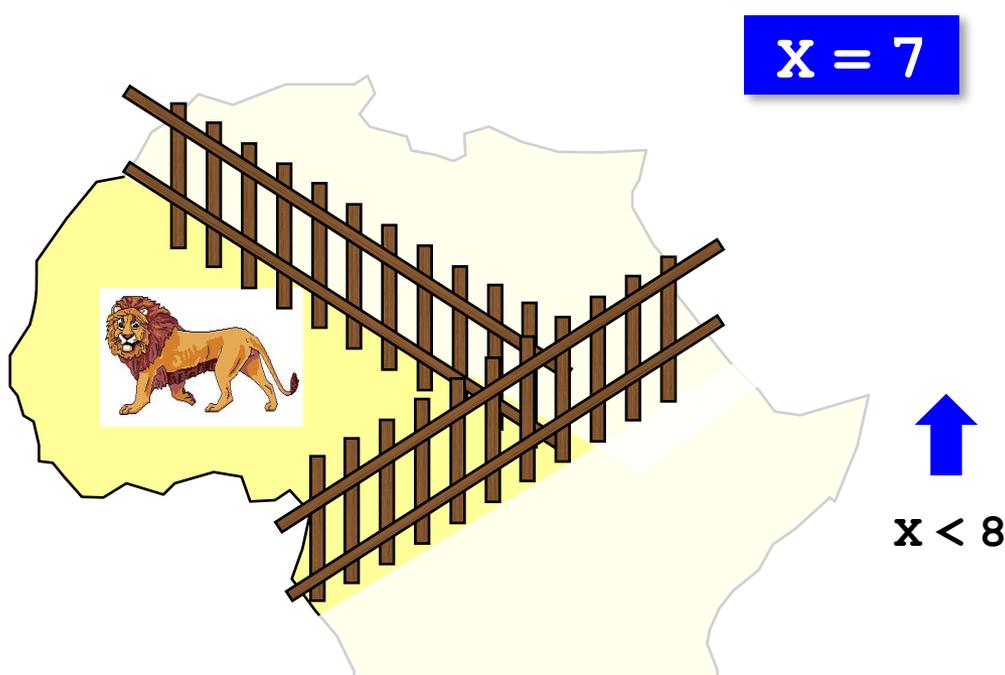
---

- «С»: Напишите программу, которая сравнивает число перестановок элементов при использовании сортировки «пузырьком», методом выбора и алгоритма быстрой сортировки. Проверьте ее на разных массивах, содержащих 1000 случайных элементов, вычислите среднее число перестановок для каждого метода.
- «D»: Попробуйте построить массив из 10 элементов, на котором алгоритм быстрой сортировки показывает худшую эффективность (наибольшее число перестановок). Сравните это количество перестановок с эффективностью метода пузырька (для того же массива).

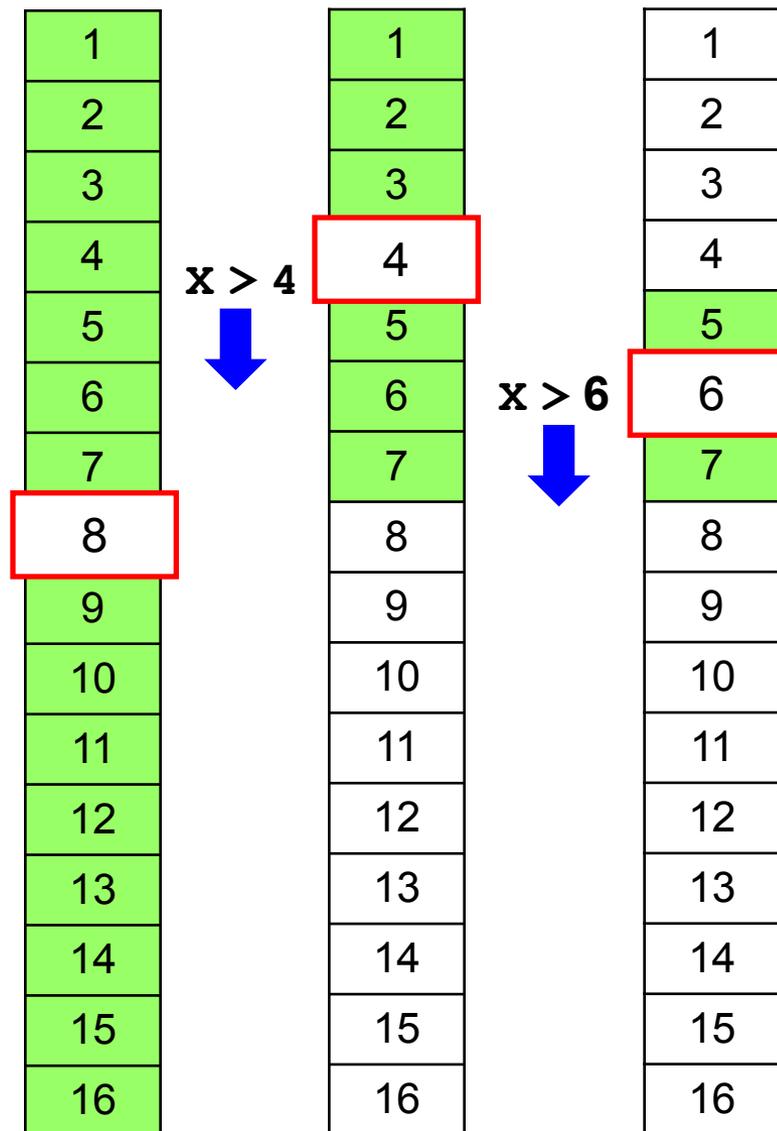
# Программирование на алгоритмическом языке

## **§ 65. Двоичный поиск**

# Двоичный поиск

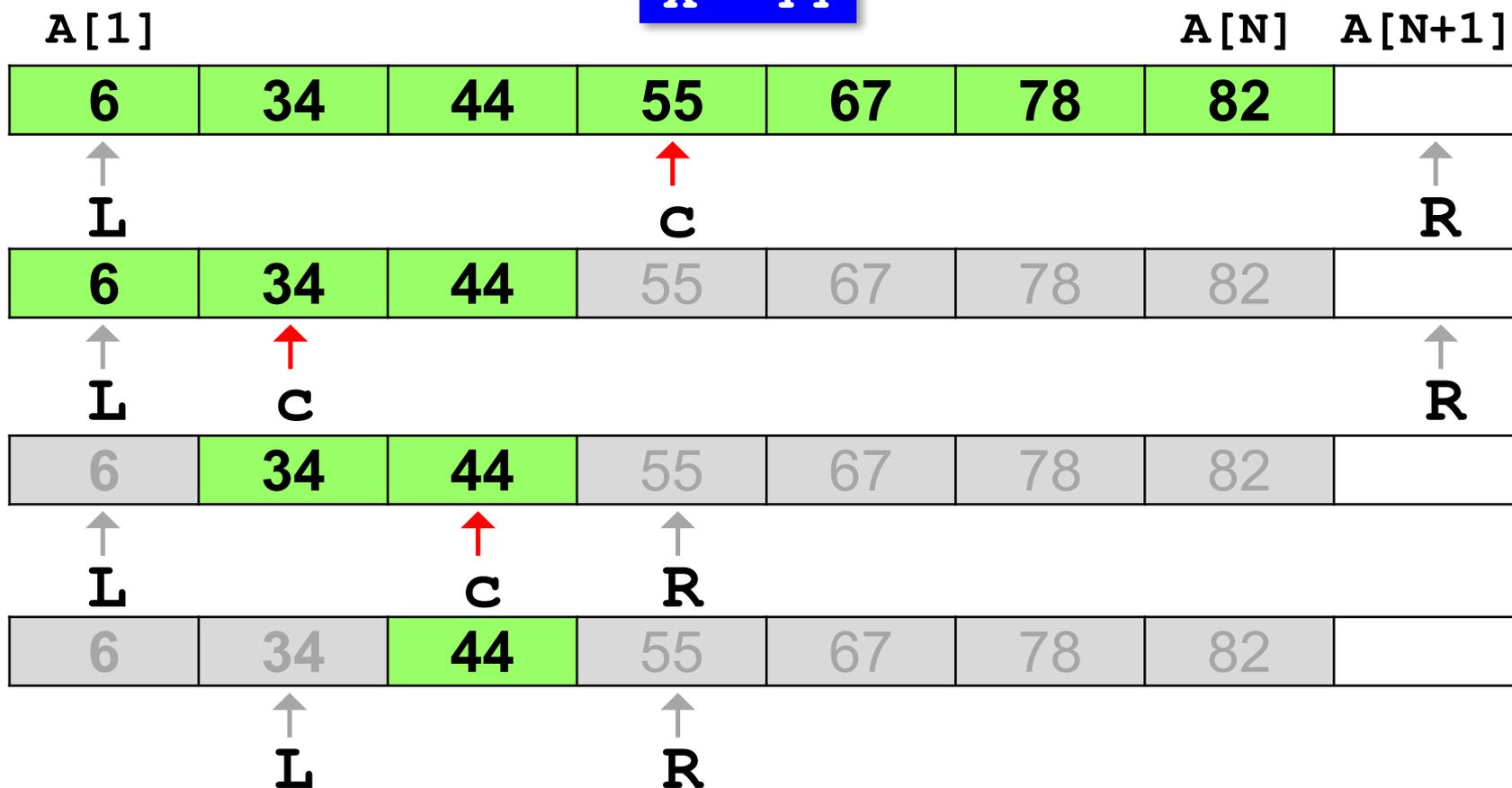


1. Выбрать средний элемент  $A[s]$  и сравнить с  $X$ .
2. Если  $X = A[s]$ , то нашли (**стоп**).
3. Если  $X < A[s]$ , искать дальше в первой половине.
4. Если  $X > A[s]$ , искать дальше во второй половине.



# Двоичный поиск

**X = 44**



**L = R - 1 : поиск завершен!**

# ДВОИЧНЫЙ ПОИСК

**цел** L, R, c

L := 1; R := N + 1 | начальный диапазон

нц пока L < R - 1

    c := div(L + R, 2) | нашли середину

    если X < A[c] то

        R := c | изменить диапазон

    иначе L := c

все

кц

если A[L] = X то

    вывод 'A[', L, '] = ', X

иначе вывод 'Не нашли!'

все

# Двоичный поиск

## Число сравнений:

N	линейный поиск	двоичный поиск
2	2	2
16	16	5
1024	1024	11
1048576	1048576	21



▪ скорость выше, чем при линейном поиске



▪ нужна предварительная сортировка



Когда нужно применять?

# Задачи

---

**«А»:** Заполнить массив случайными числами и отсортировать его. Ввести число  $X$ . Используя двоичный поиск, определить, есть ли в массиве число, равное  $X$ . Подсчитать количество сравнений.

**Пример:**

**Массив :**

1 4 7 3 9 2 4 5 2

**После сортировки:**

1 2 2 3 4 4 5 7 9

**Введите число  $X$ :**

**2**

**Число 2 найдено.**

**Количество сравнений: 2**

# Задачи

---

«В»: Заполнить массив случайными числами и отсортировать его. Ввести число  $X$ . Используя двоичный поиск, определить, сколько чисел, равных  $X$ , находится в массиве.

**Пример:**

**Массив:**

1 4 7 3 9 2 4 5 2

После сортировки:

1 2 2 3 4 4 5 7 9

Введите число  $X$ :

4

Число 4 встречается 2 раз (а) .

**Пример:**

**Массив:**

1 4 7 3 9 2 4 5 2

После сортировки:

1 2 2 3 4 4 5 7 9

Введите число  $X$ :

14

Число 14 не встречается.

# Задачи

---

«С»: Заполнить массив случайными числами и ввести число и отсортировать его. Ввести число X. Используя двоичный поиск, определить, есть ли в массиве число, равное X. Если такого числа нет, вывести число, ближайшее к X.

**Пример:**

**Массив:**

1 4 7 3 9 2 4 5 2

После сортировки:

1 2 2 3 4 4 5 12 19

Введите число X:

12

Число 12 найдено.

**Пример:**

**Массив:**

1 4 7 3 9 2 4 5 2

После сортировки:

1 2 2 3 4 4 5 12 19

Введите число X:

11

Число 11 не найдено. Ближайшее число 12.

# Программирование на алгоритмическом языке

## § 66. Символьные строки

# Зачем нужны символьные строки?

**симтаб** `s[1:80]` | **массив символов**

- ⊖ элементы массива – отдельные объекты
- сложно работать со строками переменной длины

## Хочется:

- строка – единый объект
- длина строки может меняться во время работы программы

**лит** `s` | **символьная строка**

**литерный тип**

# Символьные строки

Присваивание:

```
s := 'Вася пошёл гулять'
```

**ЛИТ** s

Ввод с клавиатуры:

```
ВВОД s
```

Вывод на экран:

```
ВЫВОД s
```



А если массив?

Отдельный символ:

```
s[4] := 'a'
```

Длина строки:

```
цел n
```

```
n := длин(s)
```

# Символьные строки

Задача: заменить в строке все буквы 'а' на буквы 'б'.

```
алг Замена а на б
нач
  лит s
  вывод 'Введите строку: '
  ввод s
  цел i
  нц для i от 1 до длин(s)
    если s[i] = 'а' то
      s[i] := 'б'
    все
  кц
  вывод s
кон
```

# Задачи

---

**«А»:** Ввести с клавиатуры символьную строку и заменить в ней все буквы «а» на «б» и все буквы «б» на «а» (заглавные на заглавные, строчные на строчные).

**Пример:**

**Введите строку:**

**ааббААББссСС**

**Результат:**

**ббааББААссСС**

# Задачи

---

**«В»:** Ввести с клавиатуры символьную строку и определить, сколько в ней слов. Словом считается последовательности непробельных символов, отделенная с двух сторон пробелами (или стоящая с краю строки). Слова могут быть разделены несколькими пробелами, в начале и в конце строки тоже могут быть пробелы.

## Пример:

Введите строку:

**Вася пошел гулять**

Найдено слов: 3

# Задачи

---

**«С»:** Ввести с клавиатуры символьную строку и найдите самое длинное слово и его длину. Словом считается последовательности непробельных символов, отделенная с двух сторон пробелами (или стоящая с краю строки). Слова могут быть разделены несколькими пробелами, в начале и в конце строки тоже могут быть пробелы.

**Пример:**

Введите строку:

**Вася пошел гулять**

Самое длинное слово: **гулять**, длина **6**

# Операции со строками

---

**Объединение (конкатенация) :**

```
s1 := 'Привет'
```

```
s2 := 'Вася'
```

```
s := s1 + ', ' + s2 + '!'
```

'Привет, Вася!'

**Срез:**

```
s := '123456789'
```

```
s1 := s[3:7] | '34567'
```

с какого символа

до какого символа

# Операции со строками

## Удаление:

```
s := '123456789'
```

```
удалить (s, 3, 6) | '129'
```

с какого  
СИМВОЛА

СКОЛЬКО  
СИМВОЛОВ

## Вставка:

```
s := '123456789'
```

```
вставить ('ABC', s, 3) | '12ABC3456789'
```

что

куда

с какого  
СИМВОЛА

# Поиск в строках

```
s := 'Здесь был Вася.'
```

что где

```
n := позиция('с', s)
```

```
если n > 0 то
```

```
    вывод 'Номер символа ', n
```

```
иначе
```

```
    вывод 'Символ не найден.'
```

```
все
```



Находит первое слева вхождение подстроки!

# Пример обработки строк

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

## Пример:

Введите имя, отчество и фамилию:

**Василий Алибабаевич Хрюндиков**

Результат:

**Хрюндиков В.А.**

Алибабаевич Хрюндиков

## 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...

Хрюндиков

Хрюндиков В.А.

# Пример обработки строк

```
алг FIO
нач
  лит s, name, name2
  цел n
  вывод 'Введите имя, отчество и фамилию'
  ввод s
  n := позиция(' ', s);
  name := s[1:n-1] | вырезать имя
  удалить(s, 1, n)
  n := позиция(' ', s)
  name2 := s[1:n-1] | вырезать отчество
  удалить(s, 1, n) | осталась фамилия
  s := s + ' ' + name[1] + '.' + name2[1] + '.'
  вывод s
кон
```

# Задачи

---

**«А»:** Ввести с клавиатуры в одну строку фамилию, имя и отчество, разделив их пробелом. Вывести фамилию и инициалы.

## Пример:

**Введите фамилию, имя и отчество:**

**Иванов Петр Семёнович**

**П.С. Иванов**

# Задачи

---

«В»: Ввести адрес файла и «разобрать» его на части, разделенные знаком ' / '. Каждую часть вывести в отдельной строке.

## Пример:

Введите адрес файла:

**C: /фото/2013/Поход/vasya.jpg**

C:

фото

2013

Поход

vasya.jpg

# Задачи

---

**«С»:** Напишите программу, которая заменяет во всей строке одну последовательность символов на другую.

**Пример:**

Введите строку:

`(X > 0) and (Y < X) and (Z > Y) and (Z <> 5)`

Что меняем: `and`

Чем заменить: `&`

Результат

`(X > 0) & (Y < X) & (Z > Y) & (Z <> 5)`

# Преобразования «строка» – «число»

## Из строки в число:

```
s := '123'
```

да или нет

```
N := лит_в_цел(s, ОК) | N = 123
```

```
если не ОК то вывод 'Ошибка!' все
```

```
s := '123.456';
```

```
X := лит_в_вещ(s, ОК) | X = 123.456
```

```
если не ОК то вывод 'Ошибка!' все
```

цел N, вещь X,  
лит s, лог ОК

## Из числа в строку:

```
N := 123
```

```
s := цел_в_лит(N) | '123'
```

```
X := 123.456
```

```
s := вещь_в_лит(X) | '123.456'
```

# Задачи

---

**«А»:** Напишите программу, которая вычисляет сумму трех чисел, введенную в форме символьной строки. Все числа целые.

**Пример:**

**Введите выражение :**

**12+3+45**

**Ответ: 60**

**«В»:** Напишите программу, которая вычисляет выражение, состоящее из трех чисел и двух знаков (допускаются только знаки «+» или «-»). Выражение вводится как символьная строка, все числа целые.

**Пример:**

**Введите выражение :**

**12-3+45**

**Ответ: 54**

# Задачи

---

**«С»:** Напишите программу, которая вычисляет выражение, состоящее из трех чисел и двух знаков (допускаются знаки «+», «-», «\*» и «/»). Выражение вводится как символьная строка, все числа целые. Операция «/» выполняется как целочисленное деление (`div`).

**Пример:**

**Введите выражение :**

**12\*3+45**

**Ответ: 81**

# Задачи

---

**«D»:** Напишите программу, которая вычисляет выражение, состоящее из трех чисел и двух знаков (допускаются знаки «+», «-», «\*» и «/») **и круглых скобок**. Выражение вводится как символьная строка, все числа целые. Операция «/» выполняется как целочисленное деление (**div**).

**Пример:**

**Введите выражение :**

**2 \* (3 + 45) + 4**

**Ответ: 100**

# Строки в процедурах и функциях

**Задача:** построить процедуру, которая заменяет в строке `s` все вхождения слова-образца `wOld` на слово-замену `wNew`.

```
нц пока | слово wOld есть в строке s  
  | удалить слово wOld из строки  
  | вставить на это место слово wNew  
кц
```



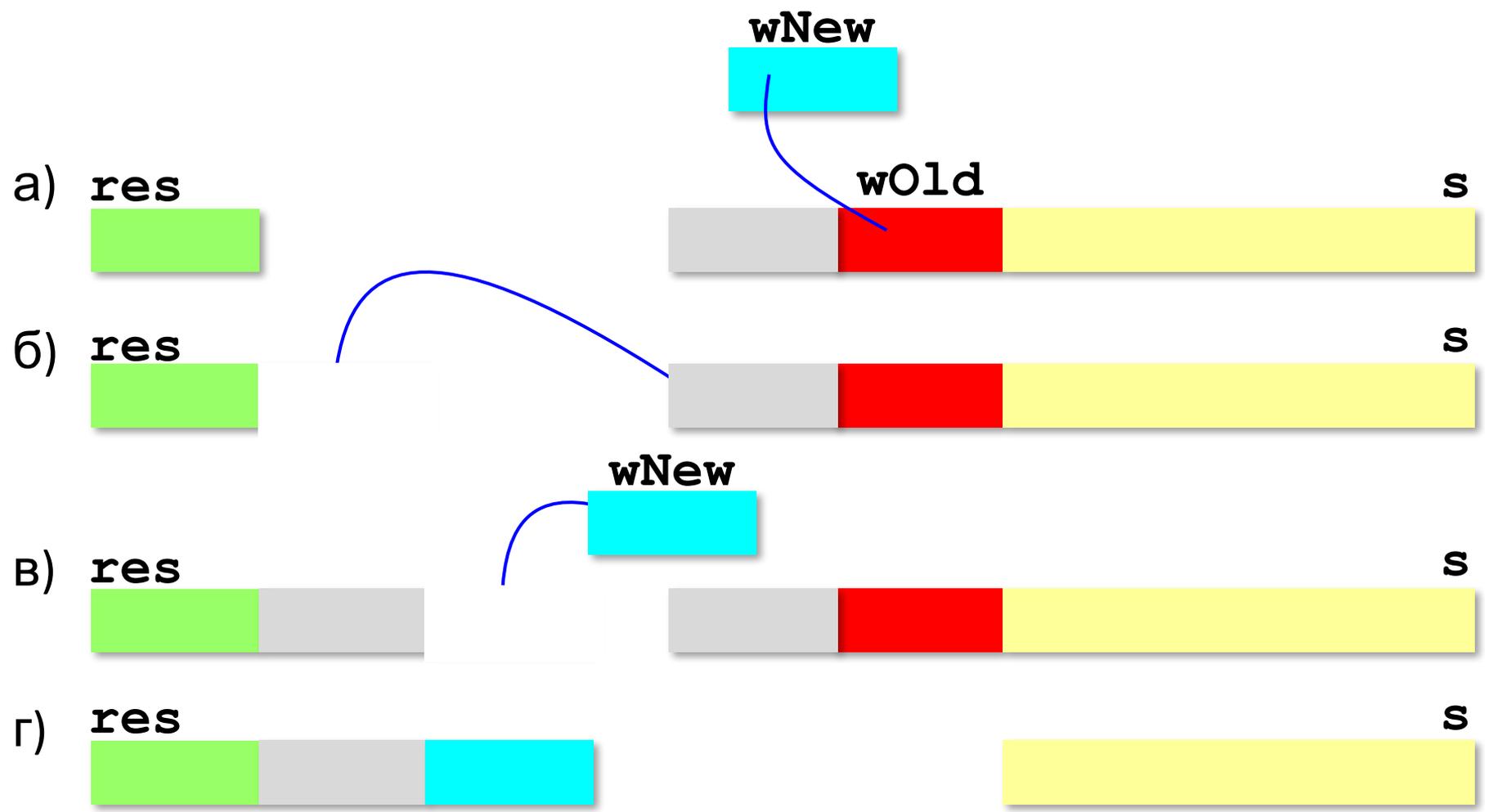
Что плохо?

wOld: '12'

wNew: 'A12B'

**зацикливание**

# Замена всех экземпляров подстроки



# Замена всех экземпляров подстроки

```
алг Замена всех
```

```
нач
```

```
  лит s = '12.12.12'
```

```
  replaceAll(s, '12', 'A12B')
```

```
  вывод s
```

```
кон
```

рабочая строка s	результат res
'12.12.12'	' '

# Замена всех экземпляров подстроки

алг `replaceAll` (аргрез **лит** `s`, арг **лит** `wOld`, `wNew`)

нач

**лит** `res`; **цел** `p`, `len`

`len` := **длин** (`wOld`)

`res` := ''

нц пока **длин** (`s`) > 0

найти слово `wOld`

`p` := **позиция** (`wOld`, `s`)

не нашли...

если `p` = 0 то `res` := `res` + `s`; **выход** **все**

если `p` > 1 то `res` := `res` + `s`[1:`p`-1] **все**

добавить то,  
что перед ним

`res` := `res` + `wNew`

добавить `wNew`

если `p`+`len` > **длин** (`s`) то

`s` := ''

иначе `s` := `s`[`p`+`len`:**длин** (`s`)]

взять «хвост»

**все**

кц

`s` := `res`

кон

# Замена: из процедуры в функцию

```
алг Замена всех
```

```
нач
```

```
  лит s = '12.12.12'
```

```
  s := replaceAll(s, '12', 'A12B')
```

```
  вывод s
```

```
кон
```

```
алг лит replaceAll(лит s0, wOld, wNew)
```

```
нач
```

```
  лит s
```

```
  s := s0
```

```
  ... | тело процедуры
```

```
  знач := res
```

```
кон
```

# Задачи

---

**«А»:** Напишите функцию, которая возвращает первое слово переданной ей символьной строки.

**Пример:**

**Введите строку:** Однажды в студёную зимнюю пору...

**Первое слово:** Однажды

# Задачи

---

«В»: Напишите функцию, которая заменяет расширение файла на заданное новое расширение.

## Пример:

Введите имя файла: `qq`

Введите новое расширение: `tmp`

Результат: `qq.tmp`

## Пример:

Введите имя файла: `qq.exe`

Введите новое расширение: `tmp`

Результат: `qq.tmp`

## Пример:

Введите имя файла: `qq.work.xml`

Введите новое расширение: `tmp`

Результат: `qq.work.tmp`

# Задачи

---

**«С»:** Напишите функцию, которая заменяет во всей строке все римские числа на соответствующие десятичные числа.

**Пример:**

Введите строку:

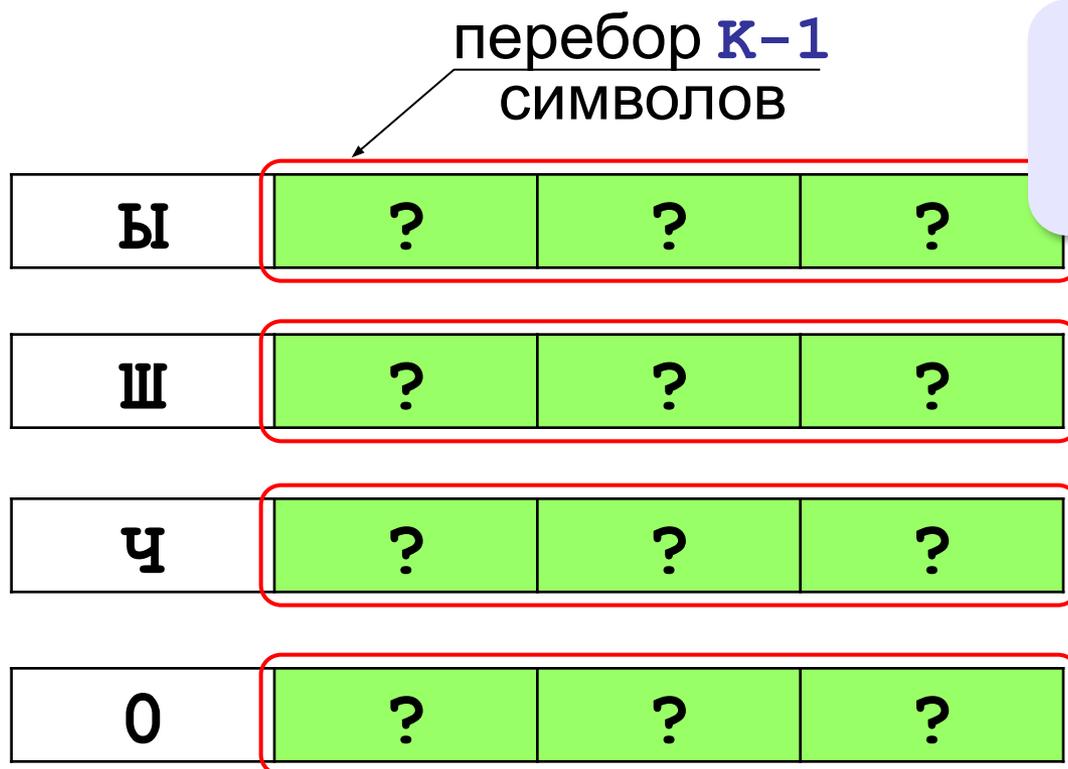
**В ММХІІІ году в школе СХХІІІ состоялся очередной выпуск ХІ классов.**

Результат:

**В 2013 году в школе 123 состоялся очередной выпуск 11 классов.**

# Рекурсивный перебор

**Задача.** В алфавите языке племени «тумба-юмба» четыре буквы: «Ы», «Ш», «Ч» и «О». Нужно вывести на экран все слова, состоящие из  $K$  букв, которые можно построить из букв этого алфавита.



задача для слов длины  $K$  сведена к задаче для слов длины  $K-1$ !

# Рекурсивный перебор

алг перебор K символов

нач

w[1] := 'Ы'

| перебор последних K-1 символов

w[1] := 'Ш'

| перебор последних K-1 символов

w[1] := 'Ч'

| перебор последних K-1 символов

w[1] := 'О'

| перебор последних K-1 символов

кон

# Рекурсивный перебор

алг **ЫШЧО**

нач

**лит** word = '...'; | **из** K **символов**

**TumbaWords** алфавит слово

кон

алг **TumbaWords** (**лит** A, w0, **цел** N)

нач

если N = **длин**(w0) то | **слово построено**

вывод w0, нс

выход

все

**цел** i; **лит** w

w := w0

нц для i от 1 до **длин**(A)

w[N+1] := A[i]

**TumbaWords**(A, w, N+1) | **рекурсия**

кц

кон

уже установлено

когда все  
символы уже  
установлены

по всем символам  
алфавита

# Задачи

---

- «А»:** В алфавите языке племени «тумба-юмба» четыре буквы: «Ы», «Ш», «Ч» и «О». Нужно вывести на экран все возможные слова, состоящие из  $K$  букв, в которых вторая буква «Ы». Подсчитайте количество таких слов.
- «В»:** В алфавите языке племени «тумба-юмба» четыре буквы: «Ы», «Ш», «Ч» и «О». Нужно вывести на экран все возможные слова, состоящие из  $K$  букв, в которых есть по крайней мере две одинаковые буквы, стоящие рядом. Подсчитайте количество таких слов.  
Программа не должна строить другие слова, не соответствующие условию.

# Задачи

---

**«С»:** В алфавите языке племени «тумба-юмба» четыре буквы: «Ы», «Ш», «Ч» и «О». Нужно вывести на экран все возможные слова, состоящие из  $K$  букв, в которых есть по крайней мере две одинаковые буквы, не обязательно стоящие рядом.

Программа не должна строить другие слова, не соответствующие условию.

# Сравнение строк

Пар ? пар ? парк

Сравнение по кодам символов:

	0	1	...	8	9
CP-1251	48	49	...	56	57
UNICODE	48	49	...	56	57
	A	B	...	Y	Z
CP-1251	65	66	...	89	90
UNICODE	65	66	...	89	90
	a	b	...	y	z
CP-1251	97	98	...	121	122
UNICODE	97	98	...	121	122

# Сравнение строк

	А	Б	...	Ё	...	Ю	Я
CP-1251	192	193	...	168	...	222	223
UNICODE	1040	1041	...	1025	...	1070	1071

	а	б	...	ё	...	ю	я
CP-1251	224	225	...	184	...	254	255
UNICODE	1072	1073	...	1105	...	1102	1103

5STEAM < STEAM < Steam < steam

steam < ПАР < Пар < пАр < пар < парк

# Сортировка строк

алг **Сортировка строк**

нач

цел  $i, j, N = 10$

литтаб  $S[1:N]$

лит  $s1$

нц для  $i$  от 1 до  $N$

ввод  $S[i]$

кц

...

нц для  $i$  от 1 до  $N$

вывод  $S[i], \text{нс}$

кц

кон

массив строк

нц для  $i$  от 1 до  $N-1$

нц для  $j$  от  $N-1$  до  $i$  шаг  $-1$

если  $S[j+1] < S[j]$  то

$s1 := S[j]$

$S[j] := S[j+1]$

$S[j+1] := s1$

все

кц

кц

# Задачи

---

**«А»:** Вводится 5 строк, в которых сначала записан порядковый номер строки с точкой, а затем – слово. Вывести слова в алфавитном порядке.

**Пример:**

**Введите 5 строк:**

- 1. тепловоз**
- 2. арбуз**
- 3. бурундук**
- 4. кефир**
- 5. урядник**

**Список слов в алфавитном порядке:**

**арбуз, бурундук, кефир, тепловоз, урядник**

# Задачи

---

**«В»:** Вводится несколько строк (не более 20), в которых сначала записан порядковый номер строки с точкой, а затем – слово. Ввод заканчивается пустой строкой. Вывести введённые слова в алфавитном порядке.

**Пример:**

**Введите слова :**

**1. тепловоз**

**2. арбуз**

**Список слов в алфавитном порядке :**

**арбуз, тепловоз**

# Задачи

---

**«С»:** Вводится несколько строк (не более 20), в которых сначала записаны инициалы и фамилии работников фирмы. Ввод заканчивается пустой строкой. Отсортировать строки в алфавитном порядке по фамилии.

## Пример:

Введите ФИО:

**А.Г. Урядников**

**Б.В. Тепловозов**

**В.Д. Арбузов**

Список в алфавитном порядке:

**В.Д. Арбузов**

**Б.В. Тепловозов**

**А.Г. Урядников**

# Программирование на алгоритмическом языке

## § 67. Матрицы

# 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.

# Объявление матриц

---

цел  $N = 3$ ,  $M = 4$

целтаб  $A[1:N, 1:M]$

вещтаб  $X[-3:0, -8:M]$

ло: строки  $1:N$ , столбцы

строки

столбцы

# Простые алгоритмы

## Заполнение случайными числами:

```
нц для i от 1 до N
  нц для j от 1 до M
    A[i,j] := irand(20,80)
    вывод A[i,j], ' '
  кц
вывод нс
кц
```



Вложенный цикл!

## Суммирование:

```
s := 0
нц для i от 1 до N
  нц для j от 1 до M
    s := s + A[i,j]
  кц
кц
```

# Задачи

---

**«А»:** Напишите программу, которая заполняет квадратную матрицу случайными числами в интервале  $[10,99]$ , и находит максимальный и минимальный элементы в матрице и их индексы.

**Пример:**

**Матрица А:**

12 14 67 45

32 87 45 63

69 45 14 11

40 12 35 15

**Максимальный элемент  $A[2,2]=87$**

**Минимальный элемент  $A[3,4]=11$**

# Задачи

---

**«В»:** Яркости пикселей рисунка закодированы числами от 0 до 255 в виде матрицы. Преобразовать рисунок в черно-белый по следующему алгоритму:

- 1) вычислить среднюю яркость пикселей по всему рисунку
- 2) все пиксели, яркость которых меньше средней, сделать черными (записать код 0), а остальные – белыми (код 255)

**Пример:**

**Матрица А:**

```
12 14 67 45
32 87 45 63
69 45 14 11
40 12 35 15
```

**Средняя яркость 37.88**

**Результат:**

```
0 0 255 255
0 255 0 255
255 255 0 0
255 0 0 0
```

# Задачи

«С»: Заполните матрицу, содержащую N строк и M столбцов, натуральными числами по спирали и змейкой, как на рисунках:

а)

1	2	3	4
10	11	12	5
9	8	7	6

б)

1	3	4	9
2	5	8	10
6	7	11	12

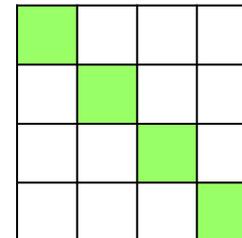
в)

1	6	7	12
2	5	8	11
3	4	9	10

# Перебор элементов матрицы

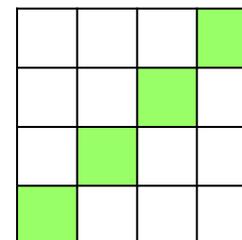
## Главная диагональ:

```
нц для i от 1 до N
  | работаем с A[i,i]
кц
```



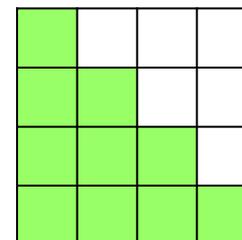
## Побочная диагональ:

```
нц для i от 1 до N
  | работаем с A[i,N+1-i]
кц
```



## Главная диагональ и под ней:

```
нц для i от 1 до N
  нц для j от 1 до i
    | работаем с A[i,j]
  кц
кц
```



# Перестановка строк

## 2-я и 4-я строки:

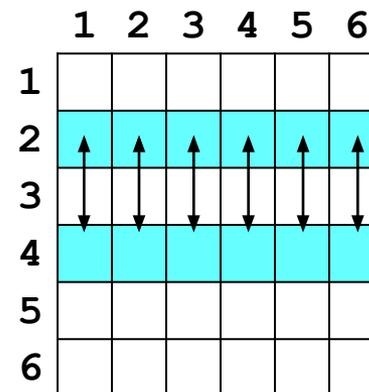
нц для  $j$  от 1 до  $M$

$c := A[2, j]$

$A[2, j] := A[4, j]$

$A[4, j] := c$

кц



# Задачи

---

**«А»:** Напишите программу, которая заполняет квадратную матрицу случайными числами в интервале  $[10,99]$ , а затем записывает нули во все элементы выше главной диагонали. Алгоритм не должен изменяться при изменении размеров матрицы.

**Пример:**

**Матрица А:**

12 14 67 45

32 87 45 63

69 45 14 30

40 12 35 65

**Результат:**

12 0 0 0

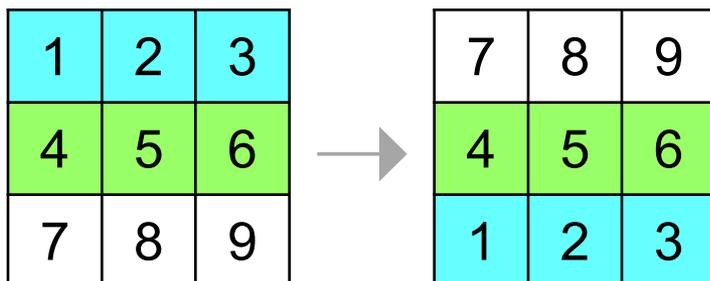
32 87 0 0

69 45 14 0

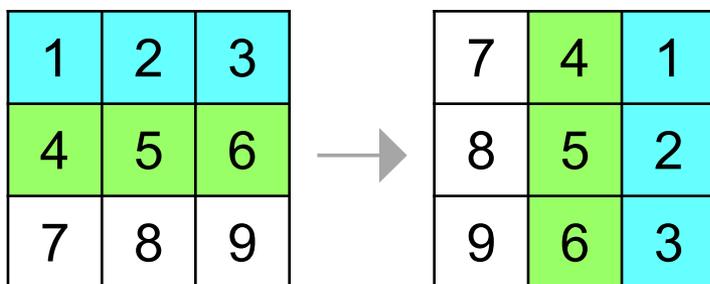
40 12 35 65

# Задачи

«В»: Пиксели рисунка закодированы числами (обозначающими цвет) в виде матрицы, содержащей N строк и M столбцов. Выполните отражение рисунка сверху вниз:



«С»: Пиксели рисунка закодированы числами (обозначающими цвет) в виде матрицы, содержащей N строк и M столбцов. Выполните поворот рисунка вправо на 90 градусов:

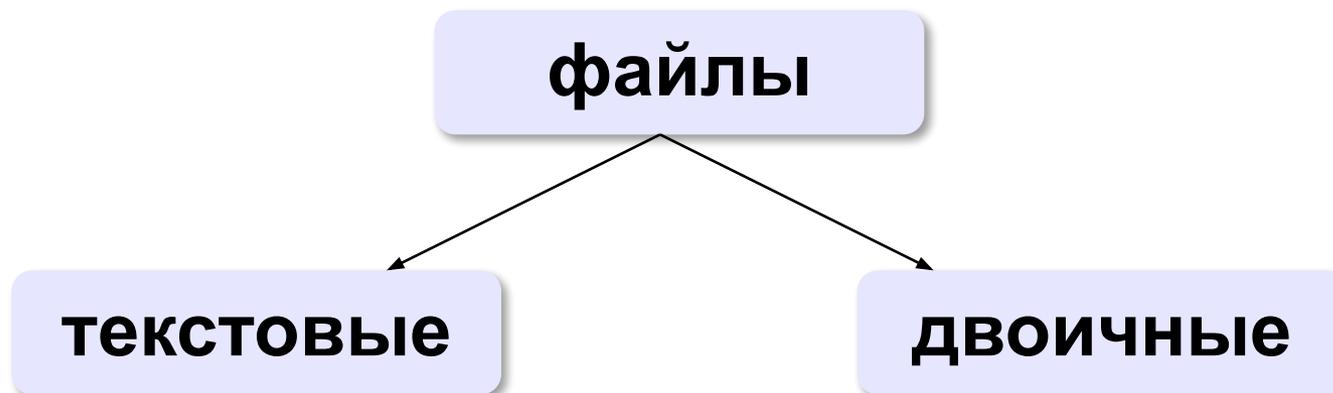


# Программирование на алгоритмическом языке

## § 68. Работа с файлами

# Как работать с файлами?

---



«*plain text*»:

- текст, разбитый на строки;
- из специальных символов только символы перехода на новую строку

- любые символы
- рисунки, звуки, видео, ...

# Принцип сэндвича



файловые  
переменные

**файл** Fin, Fout

Fin := **открыть на чтение** ('input.txt')

Fout := **открыть на запись** ('output.txt')

| **здесь работаем с файлами**

**заккрыть** (Fout)

**заккрыть** (Fin)

# Ввод данных

---

**цел** a, b

**файл** Fin

Fin := **открыть на чтение** ('input.txt')

**ввод** Fin, a, b

**закреть** (Fin)

**Переход к началу открытого файла:**

**начать чтение** (Fin)

**Определение конца файла:**

**если конец файла** (Fin)

**вывод** 'Данные кончились'

**все**

# Вывод данных в файл

---

**цел**  $a = 1, b = 2$

**файл** Fout

Fout := **открыть на запись** ('output.txt')

**вывод** Fout, a, '+', b, '=', a+b

**закреть** (Fout)

# Чтение неизвестного количества данных

**Задача.** В файле записано в столбик неизвестное количество чисел. Найти их сумму.

```
нц пока | не конец файла
  | прочитать число из файла
  | добавить его к сумме
кц
```

```
цел x, S
файл Fin
Fin := открыть на чтение ('input.txt')
S := 0
нц пока не конец файла (Fin)
  ввод Fin, x
  S := S + x
кц
закреть (Fin)
```

# Задачи

---

- «А»:** Напишите программу, которая находит среднее арифметическое всех чисел, записанных в файле в столбик, и выводит результат в другой файл.
- «В»:** Напишите программу, которая находит минимальное и максимальное среди чётных положительных чисел, записанных в файле, и выводит результат в другой файл. Учтите, что таких чисел может вообще не быть.
- «С»:** В файле в столбик записаны целые числа, сколько их – неизвестно. Напишите программу, которая определяет длину самой длинной цепочки идущих подряд одинаковых чисел и выводит результат в другой файл.

# Обработка массивов

---

**Задача.** В файле записано не более 100 целых чисел. Вывести в другой текстовый файл те же числа, отсортированные в порядке возрастания.



В чем отличие от предыдущей задачи?



Для сортировки нужно удерживать все элементы в памяти одновременно.



```
цел МАХ = 100  
целтаб А[1:МАХ]
```

# Обработка массивов

## Ввод массива:

**файл** Fin

Fin := **открыть на чтение** ('input.txt')

**цел** N

N := 0 | **счётчик прочитанных данных**

**нц** пока не **конец файла** (Fin) и **N < MAX**

    N := N + 1

**ввод** Fin, A[N]

**кц**

**закреть** (Fin)



Зачем?

# Обработка массивов

---

## Вывод результата:

**файл** Fout

Fout := **открыть на запись** ( 'output.txt' )

нц для i от 1 до N

**вывод** Fout, A[i], нс

кц

**закреть** (Fout)

# Задачи

---

- «А»: В файле записано не более 100 чисел. Отсортировать их по возрастанию последней цифры и записать в другой файл.
- «В»: В файле записано не более 100 чисел. Отсортировать их по возрастанию суммы цифр и записать в другой файл. Используйте функцию, которая вычисляет сумму цифр числа.
- «С»: В двух файлах записаны отсортированные по возрастанию массивы неизвестной длины. Объединить их и записать результат в третий файл. Полученный массив также должен быть отсортирован по возрастанию.

# Обработка строк

---

**Задача.** В файле записано данные о собаках: в каждой строчке кличка собаки, ее возраст и порода:

**Мухтар 4 немецкая овчарка**

Вывести в другой файл сведения о собаках, которым меньше 5 лет.

```
нц пока не конец файла (Fin)
  | прочитать строку из файла Fin
  | разобрать строку – выделить возраст
  если возраст < 5 то
    | записать строку в файл Fout
  все
кц
```

# Обработка строк

## Разбор строки:

- | найти в строке пробел
- | удалить из строки кличку с первым пробелом
- | найти в строке пробел
- | выделить возраст перед пробелом
- | преобразовать возраст в числовой вид

**лит** s, sAge

**цел** age, p

**лог** ОК

... | s = 'Мухтар 4 овчарка'

p := позиция(' ', s) | 'Мухтар 4 **p** овчарка'

s := s[p+1:длин(s)] | s = '4 овчарка'

p := позиция(' ', s) | '4 овчарка'

sAge := s[1:p-1] | sAge = '4'

age := лит\_в\_цел(sAge, ОК) | age = 4

p

p

# Обработка строк

```
лит s, s0
нц пока не конец файла (Fin)
  ввод Fin, s0
  s := s0
  ... | обработка строки s
  если age < 5 то
    вывод Fout, s0, нс
  все
кц
```



Зачем s0?

# Задачи

---

**«А»:** В файле записаны данные о результатах сдачи экзамена. Каждая строка содержит фамилию, имя и количество баллов, разделенные пробелами:

**<Фамилия> <Имя> <Количество баллов>**

Вывести в другой файл фамилии и имена тех учеников, которые получили больше 80 баллов.

**«В»:** В предыдущей задаче добавить к полученному списку нумерацию, сократить имя до одной буквы и поставить перед фамилией:

**П. Иванов**

**И. Петров**

...

# Задачи

---

**«С»:** В файле записаны данные о результатах сдачи экзамена. Каждая строка содержит фамилию, имя и количество баллов, разделенные пробелами:

**<Фамилия> <Имя> <Количество баллов>**

Вывести в другой файл данные учеников, которые получили больше 80 баллов. Список должен быть отсортирован по убыванию балла. Формат выходных данных:

**П. Иванов 98**

**И. Петров 96**

**...**

# Конец фильма

---

**ПОЛЯКОВ Константин Юрьевич**

д.т.н., учитель информатики

ГБОУ СОШ № 163, г. Санкт-Петербург

[kpolyakov@mail.ru](mailto:kpolyakov@mail.ru)

**ЕРЕМИН Евгений Александрович**

к.ф.-м.н., доцент кафедры мультимедийной

дидактики и ИТО ПГГПУ, г. Пермь

[eremin@pspu.ac.ru](mailto:eremin@pspu.ac.ru)

# Источники иллюстраций

---

1. [www.mcdonalds.com](http://www.mcdonalds.com)
2. иллюстрации художников издательства «Бином»
3. авторские материалы