

## **ВСЕРОССИЙСКИЕ ИНТЕРНЕТ-ОЛИМПИАДЫ ПО ИНФОРМАТИКЕ И ПРОГРАММИРОВАНИЮ**

*Санкт-Петербург по праву считается крупнейшим мировым образовательным центром в области профессиональной компьютерной подготовки студентов и школьников. Компьютерные студенческие и школьные конференции, олимпиады, конкурсы и фестивали являются важными элементами учебного процесса, направленного на пред- профессиональную компьютерную подготовку школьников. К сожалению, традиционные методы проведения творческих конкурсов по информатике и программированию не позволяют привлечь к участию в них способных ребят из регионов, удаленных от крупных образовательных центров. Например, в заключительном туре Всероссийской олимпиады школьников по информатике из-за финансовых и технологических ограничений может принять участие не более сотни ребят. Исправить создавшуюся ситуацию и существенно расширить круг участников можно только на основе применения новых подходов, к которым прежде всего относятся интернет технологии. Однако, первые опыты использования этих технологий, опирающихся, в частности, на автоматические системы тестирования программных решений, показали, что школьники, не имеющие предварительной подготовки, сталкиваются с большими психологическими трудностями. Эти трудности связаны как с новизной предъявляемых требований и задач, так и необычностью процесса проведения олимпиады.*

*Два года назад по инициативе Санкт-Петербургского государственного института точной механики и оптики (технического университета), Санкт-Петербургской межвузовской ассоциации дополнительного образования и журнала «Компьютерные инструменты в образовании» была начата работа по организации олимпиад по информатике и программированию на основе использования интернет технологий для школьников из удаленных регионов России. Всего были проведены три олимпиады, и о результатах проведения олимпиад этого года вы можете прочитать ниже. Эти олимпиады позволили обозначить наиболее острые проблемы школьной подготовки участников. В частности, даже способные ребята испытывали значительные трудности на этапах олимпиад, связанных с составлением и отладкой программ. Организаторами олимпиад был сделан вывод о необходимости существенной дополнительной подготовки школьников, которую призвана обеспечить открывающаяся Российская интернет-школа программирования. Таким образом, теперь, оценив базовый уровень своих знаний по информатике на открытых образовательных серверах тестирования [rostest.runnet.ru](http://rostest.runnet.ru) и [palm.ifmo.ru](http://palm.ifmo.ru), школьник сможет развить программистские способности в интернет-школе программирования, а эффективность обучения в школе будет оцениваться по результатам обычных и интернет-олимпиад.*

## **ВТОРАЯ ВСЕРОССИЙСКАЯ НЕДЕЛЯ «КОМПЬЮТЕРНЫЕ ТЕХНОЛОГИИ – ШКОЛЬНИКАМ РОССИИ»**

В прошлом году в Санкт-Петербурге прошла Первая Всероссийская неделя «Компьютерные технологии – школьникам России», которая получила самые положительные отзывы преподавателей и школьников. Было решено сделать эту неделю традиционной, и в этом году в начале января в Санкт-Петербургском государственном институте точной механики и оптики (техническом университете) прошел второй заключительный этап уже Второй Всероссийской недели.

В этом году в финал олимпиады вышли 107 школьников из таких мест как: Агинское (Бурятский АО), Апатиты, Армавир, Белгород, Белореченск, Заозерск, Каменск-Шахтинский, Камышин, Краснодар, Кронштадт, Кропоткин, Лангепас, Майкоп, Мурманск, Нефтеюганск, Нижневартовск, Новочеркасск, Ноябрьск, Нягань, Оленегорск, п. Осиповский Северского р-на, Полярный, Ростов-на-Дону, Снежногорск, Советск, Стрежевой, Сургут, Томская область, Ханты-Мансийск, Шахты, Югорск.

Интеллектуальные состязания молодых компьютерщиков проходили в два тура и заняли два дня – 8 и 9 января. В первый день перед ребятами выступили деканы естественно-научного факультета и факультета информационных технологий и программирования профессора С. К. Стафеев и В. Г. Парфенов. Они рассказали о новых направлениях развития современных компьютерных технологий и особенностях подготовки профессиональных компьютерщиков в ИТМО.

После лекций и беседы ребят ждало первое испытание – тестовые испытания по информатике с ограниченным объемом времени для ответов на вопросы. Итоги первого тура были представлены в виде таблицы правильных ответов на предлагаемые

вопросы и рассчитанных по ней процентов правильных решений. Более 40 ребят, показавших на первом туре лучшие результаты, были награждены сувенирами Федерации Интернет Образования.

На следующей день прошел второй тур олимпиады, на который были допущены 43 участника для решения задач по программированию и 20 участников для выполнения заданий по компьютерной графике.

Жюри второго тура олимпиады возглавляли члены сборной команды ИТМО – серебряного призера финала чемпионата мира по программированию Орландо–2000 и золотых призеров Ванкувера–2001 – студенты четвертого курса факультета информационных технологий и программирования Андрей Станкевич и Георгий Корнеев. Кстати, Георгий закончил в свое время школу в Мурманске и был приглашен для дальнейшего обучения на кафедру компьютерных технологий ИТМО за успехи на Всероссийской олимпиаде школьников по информатике. При решении задач по программированию использовалась схема интернет-олимпиад по программированию с автоматическим тестированием программных решений. Данная методика является базовой для дистанционного обучения программированию с использованием интернет технологий. Участникам были предложены четыре задачи, при решении которых они должны были выбрать алгоритм, составить и отладить программу, послать ее в жюри и добиться, чтобы она прошла все разработанные членами жюри тесты. За каждый неудачный подход участнику начислялось 20 штрафных минут. При успешной сдаче программы участнику начислялись штрафные очки, численно равные времени в минутах, прошедшему от начала состязаний до момента зачета решения. В результате напря-

женной трехчасовой борьбы три участника – Марк Васильков (Оленегорск, Мурманская область), Илларион Ишкулов (Мурманск) и Юрий Павлов (Сургут) – решили по 3 задачи и были награждены дипломами 1-й степени. Пять школьников решили по 2 задачи и получили дипломы 2-й степени и 12 школьников, решивших по 1 задаче были награждены дипломами 3-й степени.

Кроме того, по результатам состязаний по компьютерной графике жюри

наградило Илью Пшеничнова из г. Новочеркаска дипломом 2-й степени и двух школьников – Анну Золотареву из г. Каменск-Шахтинский и Дмитрия Артамонова из г. Советска – дипломами 3-й степени.

Командная победа была присуждена ребятам из Мурманской области, получившим два диплома первой степени, по одному диплому второй и третьей степеней и значительно опередившим команды остальных городов.

## **ПЕРВАЯ ВСЕРОССИЙСКАЯ ОЛИМПИАДА ПО ИНФОРМАТИКЕ ЗАТО**

Летом этого года в Санкт-Петербурге состоялась I Всероссийская олимпиада по информатике и программированию для учащихся закрытых административно-территориальных образований (ЗАТО), отдаленных военных городков и гарнизонов.

Идею Санкт-Петербургской межрегиональной ассоциации дополнительного образования (СПб МАДО) о проведении такой олимпиады поддержали Министерство образования РФ, Санкт-Петербургский Центр федерации Интернет-образования, петербургские вузы.

Цель I Всероссийской олимпиады по информатике для учащихся ЗАТО, отдаленных военных городков и гарнизонов – обеспечить этим детям равные стартовые возможности для получения высшего образования и помочь в выборе профессии.

В олимпиаде приняли участие 36 старшеклассников из Мурманской, Камчатской, Челябинской, Свердловской, Нижегородской, Московской, Калужской областей, Башкортостана, Приморского и Краснодарского края.

Участникам олимпиады прислали приветствия: В.М. Филиппов – министр образования Российской Федерации, В. А. Яковлев – губернатор Санкт-Петербурга,

Ж. И. Алферов – академик, лауреат Нобелевской премии.

Состязания проходили в три тура, задания для которых были составлены на основе примерных образовательных программ, разработанных Министерством образования РФ. В первом и втором турах проверялись знания основ информатики, в третьем туре предлагалось выполнить задания по программированию. Анализ результатов олимпиады показал, что для более успешного обучения школьников основам программирования и информатики целесообразно применять специальные педагогические методики, ориентированные на использование современных сетевых и компьютерных технологий, допускающих возможность эффективного дистанционного обучения.

Такие технологии и методики разработаны в центре подготовки программистов СПбГИТМО. Они широко используются при работе с петербургскими студентами и школьниками и положены в основу интернет-школы программирования (<http://ips.ifmo.ru>), деятельность которой осуществляется в сотрудничестве с Санкт-Петербургской межрегиональной ассоциацией дополнительного образования.

Дипломантами олимпиады стали:

**Дипломанты I степени**

1. Свириденко Максим Сергеевич, г. Обнинск;
2. Шпилев Алексей Дмитриевич, г. Железногорск;
3. Тимисков Антон Михайлович, г. Железногорск;

**Дипломанты II степени**

1. Чугаев Дмитрий Андреевич, г. Знаменск;
2. Гречишкин Александр Олегович, г. Новоуральск;
3. Семашко Александр Сергеевич, г. Дубна;

**Дипломанты III степени**

1. Зайцев Михаил Юрьевич, г. Озерск;
2. Колыхматов Илья Игоревич, г. Лесной;
3. Елисеев Всеволод Геннадьевич, г. Саров;
4. Назаров Егор Олегович, г. Озерск;

5. Платонов Кирилл Александрович, г. Фокино;

6. Попов Дмитрий Валентинович, г. Вилючинск;

7. Саврулин Роман Анатольевич, г. Озерск;

8. Сироткин Даниил Алексеевич, г. Новоуральск;

9. Короленко Геннадий Александрович, г. Межгорье;

10. Рыбак Олег Олегович, г. Сочи;

11. Теплицкий Андрей Сергеевич, г. Заозерск.

Награждение участников состязаний прошло в Смольном, дипломы победителям вручил председатель комитета по науке и высшей школе Санкт-Петербурга А.Д. Викторов.

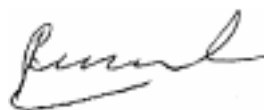
Участники олимпиады выразили желание сделать олимпиаду традиционной.

*Дорогие друзья!*

*От всей души поздравляю вас: сегодняшняя олимпиада, несомненно, станет важным событием в жизни каждого участника. Отрадно, что у талантливых детей из самых отдаленных уголков нашей Родины появилась возможность проверить собственные силы, познакомиться с ведущими вузами научной столицы России. Быть может, кому-то из вас это состязание поможет определиться в выборе будущей профессии.*

*Лучшим из вас посчастливится учиться в Петербурге. Не сомневаюсь, своим талантом и трудолюбием вы прославите наш замечательный город.*

*Бороться, искать, найти и не сдаваться!*

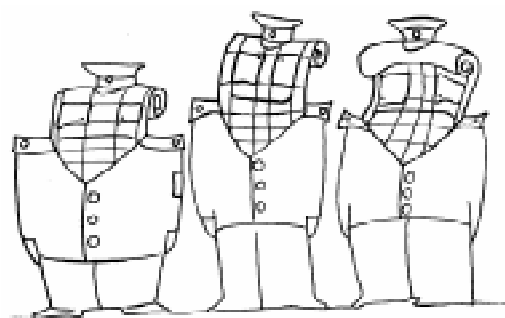


Ж.И. Алферов

## РАЗБОР ЗАДАЧ ПЕРВОЙ ВСЕРОССИЙСКОЙ ОЛИМПИАДЫ ПО ИНФОРМАТИКЕ ЗАТО

### Задача А. Матрица.

Задан целочисленный двумерный массив, состоящий из  $M$  строк и  $N$  столбцов. Написать программу, упорядочивающую его столбцы по возрастанию сумм их элементов. В случае совпадения сумм, порядок столбцов должен остаться таким, каким он был в исходном массиве.



#### Формат входных данных

Входной файл input.txt содержит числа  $M, N$  ( $1 \leq M, N \leq 100$ ) и затем  $M \times N$  целых положительных чисел, не превышающих  $10^6$ , задающих элементы массива, перечисленные по строкам (сначала  $N$  элементов первой строки, затем  $N$  элементов второй и т.д.).

#### Формат выходных данных

Выходной файл output.txt должен содержать элементы искомого массива, перечисленные по тому же принципу (размерность выводить не надо).

#### *Пример*

input.txt	output.txt
3 3	1 4 9
1 9 4	6 3 8
6 8 3	2 2 7
2 7 2	

#### **Решение.**

Задача предполагает сортировку сложных объектов (в данном случае – столбцов матрицы) по ключу (в данном

случае – по сумме элементов в столбце). Решение этой задачи не является очень сложным. Единственной проблемой, которая может возникнуть, является то, что сортировка должна быть *устойчивой* – т.е. порядок элементов с совпадающими ключами должен остаться прежним. Для решения этой проблемы можно применить один из двух методов. Первый – использовать заведомо устойчивую сортировку, такую как merge sort (сортировку слиянием). Этот метод неудобен по двум причинам – во-первых он требует аккуратного анализа сортировки (нередко даже минимальные изменения нарушают устойчивость), а во-вторых – в некоторых случаях целесообразно применение конкретной сортировки, а она в бщем случае может быть неустойчивой. Второй метод позволяет сделать любую сортировку устойчивой. Добавим дополнительный ключ – номер столбца в исходном массиве. Тогда если основной ключ у двух объектов одинаковый, будем считать меньшим в нашем порядке тот, у которого меньше дополнительный ключ (а они у всех объектов заведомо различные).

Кроме того, еще один момент состоит в том, что при сортировке больших объектов, таких как столбцы матрицы, бывает неэкономно обменивать их местами, так как, когда оценивают сложность сортировки, обычно считают, что и сравнение, и обмен происходят за константное время. В нашем случае это не так – сравнение происходит быстро (если мы заранее подсчитаем все суммы), а обмен – медленно. Поэтому будем сортировать с помощью сортировки, которая использует минимальное асимптотически количество обменов – straight selection sort (сортировку прямыми выборками). Она устроена следующим образом: сначала выбираем минимальный элемент в массиве и меняем его с первым элементом местами.

Таблица 1

```

program matrix;
var
  i, j, n, m: longint;
  min: longint;
  t: longint;
  a: array [1..100, 1..100] of longint;
  s: array [1..100] of longint;

begin
  { Открываем входной и выходной файл }
  assign(input, "input.txt");
  reset(input);
  assign(output, "output.txt");
  rewrite(output);

  { Читаем входные данные }
  read(m, n);
  for i := 1 to m do
    for j := 1 to n do
      read(a[i][j]);

  { Считаем суммы по столбцам }
  for i := 1 to n do
    begin
      s[i] := 0;
      for j := 1 to m do
        s[i] := s[i] + a[j][i];
    end;

  { Сортируем }
  for i := 1 to n do
    begin
      { Находим оставшийся столбец с минимальной суммой }
      min := i;
      for j := i + 1 to n do
        if s[j] < s[min] then
          min := j;

      { Ставим его на нужное место }
      t := s[i]; s[i] := s[min]; s[min] := t;
      for j := 1 to m do
        begin
          t := a[j][i]; a[j][i] := a[j][min]; a[j][min] := t;
        end;
    end;

  { Выводим ответ }
  for i := 1 to m do
    begin
      for j := 1 to n do
        write(a[i][j], " ");
      writeln;
    end;

  { Закрываем входной и выходной файл }
  close(input);
  close(output);
end.

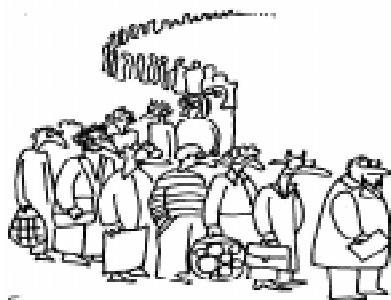
```

Затем сортируем тем же способом массив, начиная со второго элемента. Отметим, что эта сортировка является устойчивой (доказательство этого утверждения оставим в качестве несложного упражнения) и поэтому не требует заведения дополнительного ключа.

Рассмотрим программу (таблица 1). Отметим, что мы могли бы еще ускорить нашу программу, если бы не обменивали столбцы местами вообще, а меняли бы лишь указатели на столбцы. Реализацию с указателями предлагаем читателю в качестве упражнения.

### Задача В. Нетривиальный поиск.

Задана последовательность натуральных чисел. Требуется найти в ней наименьшее число, сумма цифр соседей которого – простое число (заметьте, что у первого и последнего числа по одному соседу). Если таких чисел нет, выведите «-1».



#### Формат входных данных

Входной файл input.txt содержит число  $N$  – количество чисел ( $1 \leq N \leq 100000$ ) и затем  $N$  натуральных чисел не превышающих  $10^9$ . Все числа разделены пробелами и/или переводами строк.

#### Формат выходных данных

Выходной файл output.txt должен содержать ответ на задачу.

#### Пример

input.txt	output.txt
5 4 1 4 1 2	4
4 4 4 4 4	-1

### Решение.

Отметим, что нетривиальность поиска в основном заключается в формулировке условия на то, что требуется найти. Проверка же выполнимости этого условия не очень сложна.

Рассмотрим последовательно все подзадачи, которые возникают при решении этой задачи.

**Подзадача 1.** Найти сумму цифр неотрицательного числа.

**Решение.** Рассмотрим число. Заметим, что если оно равно 0, то его сумма цифр равна 0. Теперь предположим, что число  $x$  не равно 0. Тогда его сумма цифр  $\text{digsum}(x) = (x \bmod 10) + \text{digsum}(x \div 10)$ . Тогда подсчет суммы цифр можно организовать с помощью простой функции:

```
function digsum(x: longint): longint;
var
  r: longint;
begin
  r := 0;
  while x > 0 do
  begin
    r := r + x mod 10;
    x := x div 10;
  end;
  digsum := r;
end;
```

**Подзадача 2.** Выяснить, является ли число простым.

**Решение.** Отметим, что число является простым, если оно не имеет собственных делителей, то есть делителей, отличных от 1 и его самого. Отметим также, что если число представимо в виде  $A = B \times C$ , то  $\min\{B, C\} \leq \sqrt{A}$ , так как если  $B > \sqrt{A}$  и  $C > \sqrt{A}$ , то  $B \times C > A$ , что противоречит предположению, что  $A = B \times C$ . Значит достаточно проверить, что число не имеет собственных делителей, не превышающих квадратного корня из него. Вообще говоря, достаточно проверить, что число не имеет простых делителей, не превышающих корня из него, но при этом требуется знать все соответствующие простые числа. Поскольку в нашем случае числа, с которыми мы имеем дело, неве-

лики (не превышают  $9 \times 9 \times 2 = 162$ ), то будем пользоваться более простым способом. Проверку числа на простоту реализует следующая функция:

```
function isprime(x: longint): boolean;
var
  i: longint;
begin
  i := 2;
  isprime := false;
  while i * i <= x do
  begin
    if x mod i = 0 then
      exit;
    inc(i);
  end;
  isprime := true;
end;
```

Заметим, что нам удалось избежать использования вычислений с вещественными числами, поскольку проверку на то, что число не превышает квадратного корня из заданного числа, мы заменили на проверку, что его квадрат не превышает заданного числа.

**Подзадача 3.** Найти минимум среди чисел в заданной последовательности, которые удовлетворяют нашему критерию (собственно, решить поставленную задачу).

**Решение.** Теперь, когда мы научились считать сумму цифр числа и выяснять, является ли некоторое число простым, решим поставленную задачу. Отме-

Таблица 2

```
read(n);

{ Текущее число - нулевое }
prev := 0;
cur := 0;
read(next);

{ Из входного файла прочитано 1 число }
rescv := 1;
{ Минимум пока неопределен }
min := -1;
{ Пока есть что читать }
while rescv < n do
begin
  { Сдвигаемся на 1 }
  prev := cur;
  cur := next;
  read(next);
  inc(rescv);

  { Проверяем условие, если все хорошо, то обновляем минимум }
  if isprime(sumdig(prev) + sumdig(next)) then
    if (min = -1) or (cur < min) then
      min := cur;
end;

{ Последнее число }
prev := cur;
cur := next;
next := 0;
if isprime(sumdig(prev) + sumdig(next)) then
  if (min = -1) or (cur < min) then
    min := cur;

{ Выводим ответ }
writeln(min);
```



тим, что поскольку количество чисел в нашей последовательности может быть довольно большим, нам не хотелось бы хранить их всех в памяти. В каждый момент времени будем помнить только предыдущее, текущее и следующее число. Если предыдущего или следующего числа нет, будем использовать число 0 (отметим, что сумма его цифр – как раз 0, что нас вполне устраивает).

Опуская открытие и закрытие файлов и описание переменных, рассмотрим основную программу (таблица 2).

Таким образом, задача решена.

### Задача С. Минимум.

Цифрами в системе счисления с основанием  $N \leq 36$  являются цифры от 0 до 9 и затем латинские буквы от A до Z. (Например,  $Z4_{36} = 35 \cdot 36^1 + 4 \cdot 36^0 = 1264_{10}$ ). Найдите наименьшее среди заданных в различных системах счисления чисел.



### Формат входных данных

На первой строке входного файла input.txt находится  $K$  – количество чисел

во входном файле. На следующих  $K$  строках находятся сами эти числа, в виде пары: *система счисления – число*, разделенные пробелом (система счисления задана в десятичном виде). Все числа меньше  $(10^9)_{10}$ . Явного ограничения на  $K$  нет.

### Формат выходных данных

Выведите в выходной файл наименьшее из заданных чисел в той же системе счисления, в которой оно было во входном файле (саму систему выводить не надо). Если минимальных чисел несколько, выведите первое из них.

### Пример

input.txt	output.txt
5	10
10 10	
2 10	
16 F	
3 120	
35 A	

### Решение.

В данной задаче, как и в предыдущей, требуется найти минимум в некоторой последовательности чисел, которая целиком в память не помещается. В пре-

Таблица 3

```
function digit(c: char): longint;
begin
  case c of
    "0"..'9': digit := ord(c) - ord("0");
    "A"..'Z': digit := ord(c) - ord("A") + 10;
  end;
end;

function convert(num: string; base: longint): longint;
var
  r, i: longint;
begin
  r := 0;
  i := 1;
  while i <= length(num) do
    begin
      r := r * base + digit(num[i]);
      inc(i);
    end;
  convert := r;
end;
```

дыдущей задаче сложность была в определении, принадлежит ли число множеству, в котором следует найти минимум. В данном случае проблема состоит в том, чтобы сравнить два числа.

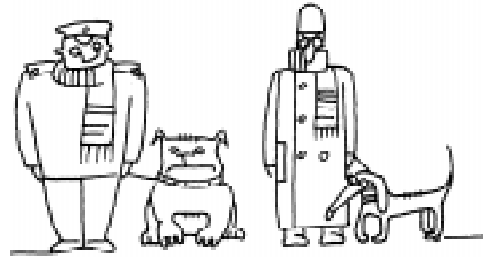
Для этого напишем функцию, конвертирующую строку, содержащую число, записанное в системе счисления с основанием  $B$  в число, представленное в стандартном целом типе данных нашего языка программирования. Обычные же числа мы можем легко сравнивать. В качестве вспомогательной напишем функцию, которая конвертирует символ, содержащий цифру в собственно цифру (таблица 3).

Теперь написание основной программы не представляет особого труда. Оставляем его в качестве упражнения.

#### Задача D. Sort Me.

Дана последовательность пар целых чисел. Отсортируйте последовательность по возрастанию первого числа. При этом

порядок пар с одинаковым первым числом должен остаться таким же, каким он был до сортировки.



#### Формат входных данных

Первая строка входного файла input.txt содержит число  $N$  – количество пар ( $1 \leq N \leq 15000$ ), следующие  $N$  строк содержат по два числа из диапазона  $[0..30000]$  каждое.

#### Формат выходных данных

Выведите в выходной файл  $N$  строк, содержащих по два числа – результат сортировки.

Таблица 4

```

procedure sort(l, r: longint);
var
  i, j, xa, xp, t: longint;
begin
  xa := a[l + random(r - l + 1)];
  xp := p[l + random(r - l + 1)];
  i := l;
  j := r;
  while i <= j do
  begin
    while (a[i] < xa) or ((a[i] = xa) and (p[i] < xp)) do
      inc(i);
    while (a[j] > xa) or ((a[j] = xa) and (p[j] > xp)) do
      dec(j);
    if i <= j then
    begin
      t := a[i]; a[i] := a[j]; a[j] := t;
      t := b[i]; b[i] := b[j]; b[j] := t;
      t := p[i]; p[i] := p[j]; p[j] := t;
      inc(i);
      dec(j);
    end;
  end;
  if l < j then
    sort(l, j);
  if i < r then
    sort(i, r);
end;

```

Пример

input.txt	output.txt
5	1 2
1 2	2 8
3 10	3 10
4 5	3 1
3 1	4 5
2 8	

**Решение.**

Как и в первой, в этой задаче требуется осуществить устойчивую сортировку объектов по ключу. В данном случае в качестве ключа выступает первое число.

Отметим, что в отличие от первой задачи, количество объектов здесь существенно больше, и сортировку с асимптотической оценкой  $O(N^2)$  применить нельзя. Среди сортировок с оценкой  $O(N \log N)$  устойчивой является уже упоминавшаяся сортировка слиянием, но она

требует  $O(N)$  дополнительной памяти и достаточно сложна в реализации. Более традиционным является применение для сортировки массивов *сортировки Хоара* или *быстрой сортировки* (quicksort), но она при стандартной реализации не является устойчивой. Однако, как уже упоминалось в разборе первой задачи, любая сортировка может быть сделана устойчивой при применении дополнительного ключа – позиции в исходном массиве. Применим быструю сортировку с указанной модификацией к нашей задаче. Кроме того, поскольку быстрая сортировка в худшем случае выполняет  $O(N^2)$  действий, модифицируем ее, добавив случайность в выбор среднего элемента на каждой итерации.

Предположим, что массив  $a$  содержит первый элемент пары, массив  $b$  – второй, а массив  $p$  – позицию объекта в исходном массиве. Тогда процедура сортировки выглядит, как показано в таблице 4.



*Станкевич Андрей Сергеевич,  
член оргкомитета  
Всероссийской интернет-олимпиады  
по информатике и программированию.  
Харченко Татьяна Владимировна,  
член оргкомитета  
Всероссийской интернет-олимпиады  
по информатике и программированию.*