Д. М. Златопольский,
школа № 1530 «Школа Ломоносова», г. Москва

МЕТОДИКА ВЫПОЛНЕНИЯ ЗАДАНИЯ 27 ДЕМОНСТРАЦИОННОГО ВАРИАНТА ЕГЭ ПО ИНФОРМАТИКЕ И ИКТ 2019 ГОДА, ИЛИ «НЕ ТАК СТРАШЕН ЧЕРТ...»

Аннотация
В статье подробно и доступно описывается методика выполнения самого сложного задания по программированию из демонстрационного варианта ЕГЭ по информатике. При описании используется школьный алгоритмический язык (система «КуМир»), что делает программы максимально понятными и легко переносимыми на любой другой язык программирования.

***Ключевые слова**:* информатика, ЕГЭ, программирование, методика решения.
DOI: 10.32517/2221-1993-2018-17-8-25-27

Контактная информация

Златопольский Дмитрий Михайлович, канд. тех. наук, доцент, школа № 1530 «Школа Ломоносова», г. Москва; *адрес:* 107014, г. Москва, ул. Егерская, д. 4; *телефон:* (495) 603-09-62; *e-mail:* zlatonew@gmail.com

D. M. Zlatopolski,
School 1530 «School of Lomonosov», Moscow

THE METHOD OF SOLVING THE TASK 27 OF THE DEMONSTRATION VERSION OF THE UNIFIED STATE EXAM ON INFORMATICS 2019, OR "NOT SO TERRIBLE DEVIL..."

Abstract
The article describes in detail the method of solving the most difficult task on programming from the demonstration version of the Unified State Exam on informatics 2019. In the description, the school algorithmic language is used (the KuMir system), which makes programs understandable and easily portable to any other programming language.

Keywords: informatics, Unified State Exam, programming, method of solving.

Как известно, самым сложным заданием по программированию на ЕГЭ по информатике и ИКТ является задание 27. Обсудим вариант этого задания, предложенный в демоверсии ЕГЭ 2019 года [1].

Задание.

На вход программы поступает последовательность из N целых положительных чисел, все числа в последовательности различны. Рассматриваются все пары различных элементов последовательности, находящихся на расстоянии не меньше чем 4 (разница в индексах элементов пары должна быть 4 или более, порядок элементов в паре неважен). Необходимо определить количество таких пар, для которых произведение элементов делится на 29.

Описание входных и выходных данных.

В первой строке входных данных задается количество чисел N ($4 \le N \le 1000$). В каждой из последующих N строк записано одно целое положительное число, не превышающее 10 000.

В качестве результата программа должна вывести одно число: количество пар элементов, находящихся на расстоянии не меньше чем 4, в которых произведение элементов кратно 29.

Пример входных данных:

```
7
58
2
3
5
4
1
29
```

Пример выходных данных для приведенного выше примера входных данных:

```
5
```

Пояснение. Из семи заданных элементов с учетом допустимых расстояний между ними можно составить шесть произведений: $58 \cdot 4$, $58 \cdot 1$, $58 \cdot 29$, $2 \cdot 1$, $2 \cdot 29$, $3 \cdot 39$. Из них на 29 делятся пять произведений.

Требуется написать эффективную по времени и по памяти программу для решения описанной задачи.

Программа считается эффективной по времени, если при увеличении количества исходных чисел N в k раз время работы программы увеличивается не более чем в k раз.

Программа считается эффективной по памяти, если память, необходимая для хранения всех переменных программы, не превышает 1 килобайта и не увеличивается с ростом N.

Максимальная оценка за правильную (не содержащую синтаксических ошибок и дающую правильный ответ при любых допустимых входных данных) программу, эффективную по времени и по памяти, — 4 балла.

Максимальная оценка за правильную программу, эффективную только по времени, — 3 балла.

Максимальная оценка за правильную программу, не удовлетворяющую требованиям эффективности, — 2 балла.

Вы можете сдать одну программу или две программы решения задачи (например, одна из программ может быть менее эффективна). Если вы сдадите две программы, то каждая из них будет оцениваться независимо от другой, итоговой станет **бóльшая** из двух оценок.

Перед текстом программы обязательно кратко опишите алгоритм решения.

Укажите использованный язык программирования и его версию.

Решение.

При разработке программ будем использовать школьный алгоритмический язык (система «КуМир»). Русский синтаксис этого языка делает программу максимально понятной и легко переносимой на любой другой язык программирования.

Обсуждаемая задача может быть решена путем полного перебора всех пар элементов (в данном случае — отстоящих друг от друга не менее чем на четыре элемента).

Идея решения: записываем все входные значения *a* в массив *m**, *после чего рассматриваем все пары элементов — если произведение их значений кратно 29, то увеличиваем счетчик искомых значений (обозначим его k*29) на 1.

Соответствующая программа:

```
алг Задание_27
нач цел N, a, i, j, k29, цел таб m[1:1000]
  ввод N
  |Ввод значений и запись их в массив
  нц для i от 1 до N
    ввод a
    m[i] := a
  кц
  k29 := 0
  нц для i от 1 до N - 4
    нц для j от i + 4 до N
      если mod(m[i] * m[j], 29) = 0
        то
         k29 := k29 + 1
      все
    кц
  кц
  вывод нс, k29
кон
```

Такое решение не является эффективным ни по памяти (используется массив, размер которого зависит от значения *N*), ни по времени (происходит рассмотрение и проверка большого числа пар элементов, количество которых также зависит от количества исходных чисел *N*).

Попробуем обрабатывать числа по мере записи их в массив. Будем учитывать, что произведение двух чисел делится на 29, если хотя бы одно из этих чисел делится на 29.

Рассмотрим некоторый очередной, например 10-й, элемент массива.

* Можно переменную *a* отдельно не вводить, а сразу записывать вводимые числа в массив.

Если его значение кратно 29:

4	87	8	13	58	29	116	56	23	**145**
1	2	3	4	5	6	7	8	9	10

то искомое значение *k*29 увеличится на общее количество чисел, отстоящих от 10-го на четыре и более элементов. Количество таких чисел равно шести (10 – 4).

Если значение 10-го элемента массива не кратно 29:

4	87	8	13	58	29	116	56	23	**146**
1	2	3	4	5	6	7	8	9	10

то искомое значение *k*29 увеличится на количество чисел, кратных 29, среди элементов, отстоящих от 10-го на четыре и более элементов. В приведенном примере таких чисел три (87, 58 и 29). Обратим внимание на то, что среди них должен быть учтен и элемент, отстоящий от 10-го на четыре позиции (29). Это значит, что 6-й элемент следует проверить до проверки 10-го. Итак, мы пришли к необходимости подсчета количества элементов, кратных 29, находящихся по мере ввода значений элементов массива от очередного на четыре и более элементов. Обозначим соответствующую переменную *k*29*_ранее*.

Соответствующий фрагмент для некоторого очередного *i*-го элемента:

```
…
|Ввод значения очередного элемента массива
ввод m[i]
|Проверка (i - 4)-го элемента
если mod(m[i - 4], 29) = 0
  то
    k29_ранее := k29_ранее + 1
все
|Проверка очередного элемента
если mod(m[i], 29) = 0
  то
    k29 := k29 + (i – 4)
  иначе
    k29 := k29 + k29_ранее
все
```

Прежде чем представлять всю программу решения задачи, заметим, что так как должны обрабатываться все числа, начиная с 5-го:

```
нц для i от 5 до N
  |Ввод и обработка очередного числа
  ввод очер
  …
```

то первые четыре числа должны быть уже записаны в массив *m* до этого.

Итак, программа:

```
алг Задание_27
нач цел таб m[1:1000], цел N, k29, k29_ранее, i
  |Ввод общего количества чисел
  ввод N
  |Ввод первых четырех чисел
  нц для i от 1 до 4
        ввод m[i]
  кц
  |Начальные значения используемых величин
    k29_ранее := 0
    k29 := 0
  |Ввод и обработка остальных чисел
  нц для i от 5 до N
        ввод m[i]
    если mod(m[i - 4], 29) = 0
```

```
      то
        k29_ранее := k29_ранее + 1
    все
    |Проверка очередного элемента
    если mod(a[i], 29) = 0
      то
        k29 := k29 + (i - 4)
      иначе
        k29 := k29 + k29_ранее
    все
  кц
  |Вывод ответа
     вывод нс, k29
кон
```

Примечание. В [1] вместо конкретного значения 4 используется константа *s* (соответственно, вместо значения 5 используется *s* + 1).

Приведенная программа эффективна по времени и неэффективна по памяти (по-прежнему используется массив, размер которого зависит от *N*). Указанный недостаток можно устранить, учитывая, что при обработке очередного числа проверяется число, отстоящее от очередного на четыре «позиции». Значит, достаточно хранить в массиве только четыре последних числа. Если имя этого массива принять также за *m*, а очередное число обозначить как *очер*, то фрагмент программы, связанный с вводом и обработкой очередного, *i*-го, числа, оформляется так:

```
…
|Ввод очередного числа
ввод очер
|Проверка 1-го элемента массива
|(отстоящего от очередного на 4 «позиции»
если mod(m[1], 29) = 0
  то
    k29_ранее := k29_ранее + 1
все
|Проверка очередного числа
если mod(очер, 29) = 0
  то
    k29 := k29 + (i - 4)
  иначе
    k29 := k29 + k29_ранее
все
```

Понятно, что первые четыре числа также должны быть предварительно записаны в массив *m*.

После ввода и обработки каждого очередного числа этот массив должен быть изменен. Как? Если, например, массив выглядел следующим образом:

8	13	58	29
1	2	3	4

а очередной элемент был равен 116, то массив должен стать таким:

13	58	29	116
1	2	3	4

Обсудим эту частную задачу.

Ясно, что сразу записать в программе *m*[4] := 116 нельзя (исходное значение *m*[4] будет утеряно). Сначала следует провести сдвиг элементов влево, начиная со второго:

```
m[1] := m[2]
m[2] := m[3]
…
m[3] := m[4]
```

Если индекс элементов слева от знака присваивания обозначить *i*, то можно использовать оператор цикла с параметром:

```
нц для i от 1 до 3
  m[i] := m[i + 1]
кц
|Запись нового числа в конец массива
m[4] := 116
Можно также сдвиг элементов провести так:
нц для i от 2 до 4
  m[i - 1] := m[i]
кц
```

Вся основная программа, которая является эффективной и по памяти, и по времени, имеет вид:

```
алг Задание_27
нач цел таб m[1:4], цел N, очер, k29, k29_ранее, i, j
  |Ввод общего количества чисел
  ввод N
  |Ввод первых четырех чисел
  нц для i от 1 до 4
        ввод m[i]
  кц
  |Начальные значения используемых величин
    k29_ранее := 0
    k29 := 0
  |Ввод и обработка остальных чисел
  нц для i от 5 до N
        ввод очер
    если mod(m[1], 29) = 0
      то
        k29_ранее := k29_ранее + 1
    все
    |Проверка очередного числа
    если mod(очер, 29) = 0
      то
        k29 := k29 + (i - 4)
      иначе
        k29 := k29 + k29_ранее
    все
    |Сдвиг элементов массива влево
     нц для j от 1 до 3 |Индекс i использовать нельзя
      m[j] := m[j + 1]
    кц
    |Запись нового (текущего) числа в конец массива
    m[4] := очер
  кц
  |Вывод ответа
     вывод нс, k29
кон
```

Примечание. Можно вместо использования массива из четырех элементов хранить четыре числа в четырех «обычных» переменных, также меняя их значения аналогично описанному. Конечно, когда величина «расстояния» между учитываемыми числами большая, целесообразно использовать массив.

В заключение заметим, что все три описанные варианта программы применимы для решения аналогичных задач — когда требуется определить количество таких пар чисел, отстоящих в последовательности на заданное «расстояние», у которых произведение элементов кратно некоторому числу, являющемуся простым.

Список использованных источников

1. Демонстрационный вариант контрольных измерительных материалов Единого государственного экзамена 2019 года по информатике и ИКТ // Демоверсии, спецификации, кодификаторы. Федеральный институт педагогических измерений. http://www.fipi.ru/sites/default/files/document/1535628955/inf_.zip