

# «Мир информатики»

Журнал для тех, у кого информатика – любимый школьный предмет

Выпуск № 70, сентябрь 2022 г.

## Школа программирования

*Когда человек хочет передвинуть гору,  
он начинает с того, что убирает маленькие камни.*  
Восточная мудрость

В следующих выпусках нашего журнала будут приведены программы, моделирующие несколько вариантов решения известной головоломки «Ханойская башня», в которой нужно перенести диски с одного стержня на другой, используя третий стержень как вспомогательный и соблюдая ряд правил. В данном выпуске мы обсудим ряд вопросов, которые понадобятся при разработке указанных программ. Конечно, рассмотренные здесь вопросы будут полезны начинающим программистам вне зависимости от решения головоломки «Ханойская башня».

### Об использовании вспомогательных процедур

В программах, кроме стандартных (предусмотренных в языке программирования) функций и процедур<sup>1</sup>, могут использоваться и нестандартные, созданные пользователем, функции и процедуры.

О нестандартных вариантах функций и процедур можно сказать, что это – фрагмент программы, в котором решается какая-то часть всей задачи. Этот фрагмент оформляется особым образом и снабжается именем. Возникает вопрос: «Зачем нужно какую-то часть программы оформлять отдельно?». Чтобы ответить на него, рассмотрим несколько примеров.

В программах часто встречаются повторяющиеся или похожие фрагменты. Например, для того, чтобы вывести на экран «заставку» с изображением, представленным на рис. 1,

```
*****  
ЭТУ ПРОГРАММУ РАЗРАБОТАЛ МИТЯ ХАКЕРОВ  
*****
```

Рис. 1

– в программе на школьном алгоритмическом языке<sup>2</sup> должно быть записано:

```
нц для i от 1 до 40  
  вывод "*" "  
кц  
вывод нс  
нц для i от 1 до 10
```

<sup>1</sup> В языках программирования Си, Python и некоторых других понятия «функция» и «процедура» называются общим термином – «функция».

<sup>2</sup> Как принято в нашем журнале, мы используем школьный алгоритмический язык (система программирования КуМир). Русский синтаксис этого языка делает приводимые методики и программы максимально понятными, и вы сможете разработать аналогичные программы на языке программирования, который изучаете.

```

вывод " "
кц
вывод нс, "ЭТУ ПРОГРАММУ РАЗРАБОТАЛ МИТЯ ХАКЕРОВ"
вывод нс
нц для i от 1 до 40
  вывод "*"
кц

```

Видно, что в программе есть два абсолютно одинаковых фрагмента, относящиеся к выводу линий из звездочек. Можно эти фрагменты оформить как процедуру.

На школьном алгоритмическом языке общий вид процедуры следующий<sup>3</sup>:

```

алг <имя процедуры>
нач <описание переменных величин, используемых в процедуре>
  <тело процедуры>
кон

```

где <описание переменных величин, используемых в процедуре> – перечень используемых в процедуре величин с указанием их типа; <тело процедуры> – операторы, реализующую решаемую с помощью процедуры задачу.

В нашей задаче процедура, с помощью которой можно получить линию из 40 звездочек, имеет вид:

```

алг Линия
нач цел i
  нц для i от 1 до 40
    вывод "*"
  кц
кон

```

Созданную процедуру можно использовать в основной части программы. Вызов процедуры на выполнение осуществляется отдельным оператором, в котором указывается имя процедуры<sup>4</sup>:

```

Линия
вывод нс
нц для i от 1 до 10
  вывод " "
кц
вывод нс, "ЭТУ ПРОГРАММУ РАЗРАБОТАЛ МИТЯ ХАКЕРОВ"
вывод нс
Линия

```

Нетрудно убедиться, что в данном случае применение процедуры уменьшает размер программы (это преимущество проявилось бы в еще большей степени, если бы в программе пришлось рисовать линии три и более раз).

Еще один пример – пусть в значительной степени условный, но достаточно показательный. Если необходимо с помощью исполнителя **Чертежник** нарисовать из линий изображение слова «МУ», то соответствующая программа на школьном алгоритмическом языке оформляется так:

<sup>3</sup> С правилами оформления процедур в языке программирования, который вы изучаете, ознакомьтесь самостоятельно.

<sup>4</sup> Такие операторы называют «операторами вызова процедуры».

```

алг МУ
нач
...
  сместиться на вектор(0, 100)
  сместиться на вектор(50, -50)
  сместиться на вектор(50, 50)
  сместиться на вектор(0, -100)
  поднять перо
  сместиться на вектор(10, 0)
  опустить перо
  сместиться на вектор(50, 0)
  сместиться на вектор(0, 100)
...
кон

```

Легко ли в такой программе найти ошибочный оператор, из-за которого, например, буква У изображена неправильно? Где нужно добавить операторы, чтобы получить на экране вместо «МУ» – слово «МЯУ»?

Если же рисование букв М и У оформить в виде отдельных процедур с соответствующими именами, то основная часть программы примет вид:

```

алг МУ
нач
  сместиться в точку(50, 100)
  М | Вызов процедуры М
  сместиться в точку(160, 100)
  У | Вызов процедуры У
кон

```

Конечно, при этом размер всей программы несколько увеличится, но зато проявятся другие преимущества – программа станет более понятной, в ней легко найти нужное место, в том числе и связанное с ошибками. Если же нужно изобразить на экране слово МУМУ, то тогда применение процедур, кроме того, приведет и к уменьшению размера программы.

Возможность использования процедур и функций позволяет применять современные методы проектирования программ. Дело в том, что при программировании достаточно сложной задачи решить ее «одним махом», т. е. написать последовательно всю программу от начала до конца, обычно невозможно. Процесс разработки программ – это творческий процесс, проходящий в несколько этапов. Вначале стараются разработать наиболее общую схему алгоритма, не останавливаясь на технических деталях его реализации. В результате такой алгоритм представляется в виде последовательности относительно крупных блоков, реализующих более или менее самостоятельные смысловые части алгоритма, в которых решаются какие-то частные задачи. Эти блоки, в свою очередь, могут разбиваться на меньшие подблоки и т. д. Процесс последовательного структурирования программы продолжается до тех пор, пока реализуемые блоками алгоритмы не станут простыми и легко программируемыми.

Рассказав о преимуществах применения процедур, пойдем дальше.

Возможно также использование так называемых «процедур с параметрами» – процедур, результат выполнения которых зависит от некоторых величин – их параметров. Пусть, например, в программе требуется рисовать линии из звездочек разной «длины»:

```

...
нц для i от 1 до 40
  вывод "*"
кц
...
нц для i от 1 до 20
  вывод "*"
кц

```

```

...
нц для i от 1 до 30
  вывод "*"
кц

```

Можно, конечно, для решения такой задачи создать три отдельные процедуры, решающие эти задачи. Но выиграем ли мы при этом в размере программы? Нет, конечно. Желательно разработать процедуру, которая «умеет» рисовать линии из звездочек *любой* длины. Это и будет упомянутая чуть выше процедура с параметром.

Общий вид таких процедур на школьном алгоритмическом языке:

```

алг <имя процедуры> (<список формальных параметров>)
нач <описание переменных величин, используемых в процедуре>
  <тело процедуры>
кон

```

где <список формальных параметров> – перечень величин, от которых зависит результат выполнения процедуры (с указанием их типа); эти величины называют «формальными параметрами процедуры». Формальные параметры используются в теле процедуры.

В нашем случае процедура, с помощью которой можно получить линию любой «длины», имеет вид:

```

алг Линия (цел k)
нач цел i
  нц для i от 1 до k
    вывод "*"
  кц
кон

```

Результат её выполнения зависит от величины k, которая и является единственным параметром процедуры.

Вызов процедуры с параметрами осуществляется так:

```

<имя процедуры> (<список фактических параметров>)

```

где <список фактических параметров> – перечень конкретных значений параметров процедуры для решения конкретной задачи. Так, в нашем случае, для того чтобы использовать процедуру для рисования линии из 30 звездочек, вызов процедуры должен быть оформлен следующим образом:

```

Линия (30)

```

В качестве значений фактических параметров можно указывать<sup>5</sup>:

- 1) константы (заданные, известные значения) – именно так было сделано в приведенном примере;
- 2) имена величин, например, Линия(длина1); как видно из примера, имена фактических и формальных параметров не обязательно должны совпадать;
- 3) выражения (Линия(длина1 + 10)).

В каждом из этих трех случаев тип фактического параметра должен совпадать с типом формального параметра, указанным в заголовке процедуры.

Ясно, что в процедуре могут быть использованы два и более формальных параметра. В таких случаях при оформлении списка фактических параметров следует учитывать следующие требования:

- 1) количество фактических параметров должно быть таким же, как и в списке формальных параметров в описании процедуры;
- 2) тип каждого фактического параметра должен совпадать с типом соответствующего формального параметра;

---

<sup>5</sup> Строго говоря, это не всегда так, но в данной статье мы не будем описывать другие варианты.

3) соответствующие фактические и формальные параметра должны совпадать по смыслу.

Если первые два требования ясны и понятны, то третье требует комментариев. Пусть, например, подготовлена процедура, изображающая на экране прямоугольник:

```
алг Линия (цел а, b)
нач
...
кон
```

Как с её помощью нарисовать прямоугольник шириной 20 и высотой 100? Какой из вариантов оформления вызова процедуры является правильным: `Линия (20, 100)` или `Линия (100, 20)`? Ответ зависит от того, какому размеру соответствует формальный параметр `а`, а какому – `б`. Если `а` – это высота прямоугольника, то правильный вариант – `Линия (100, 20)`, в противном случае первой должна быть указана ширина изображаемого прямоугольника, а второй – высота (`Линия (20, 100)`).

Важно понимать, что происходит в памяти компьютера при использовании процедур (или функций). Когда программа запускается на выполнение, она размещается в оперативной памяти. Это можно смоделировать в виде прямоугольника; в этом прямоугольнике записаны операторы программы:

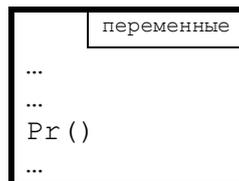


Рис. 2

Предусмотрено также место для размещения значений всех переменных программы.

Операторы программы начинают выполняться. Когда доходит очередь до оператора вызова процедуры `Pr ()`, то:

- 1) выполнение основной части программы приостанавливается;
- 2) в памяти отводится место для размещения данной процедуры. Это также можно смоделировать в виде прямоугольника с текстом процедуры:

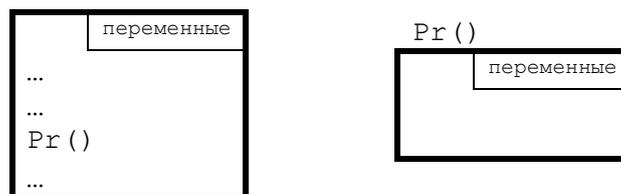


Рис. 3

В «новом» прямоугольнике также предусмотрено место для размещения значений всех переменных, использованных в процедуре;

- 3) выполняются операторы процедуры;
- 4) после выполнения всех операторов процедуры место в памяти для нее освобождается:

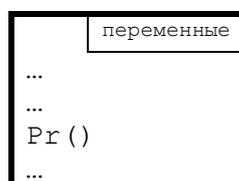


Рис.4

После этого продолжается выполнение операторов основной части программы. Если в программе еще раз встречается вызов процедуры – происходит то же самое.

Заметим, что переменные величины, описанные в процедуре или функции, называют «локальными», а описанные в основной части программы – «глобальными»<sup>6</sup>.

Из приведенного порядка работы программы при использовании процедур и функций следует, что локальные переменные «живут» только во время выполнения «своих» процедур и функций, и поэтому их значения недоступны в основной части программы и в других процедурах или функциях. Глобальные же переменные «внутри» процедур и функций можно использовать.

Например, при выполнении следующих двух программ появится сообщение об ошибке, связанное с этим обстоятельством:

```
1.
алг Основная часть программы7
нач
    Процедура1
    вывод а
кон
алг Процедура1
нач цел а
    а := 100
кон
```

```
2.
алг Основная часть программы
нач
    Процедура2
    Процедура1
кон
алг Процедура2
нач цел а
    а := 100
кон
алг Процедура1
нач
    вывод а
кон
```

В то же время глобальные переменные доступны («видны») в процедурах:

```
цел а | Глобальная переменная
алг Основная часть программы
нач
    а := 100
    Процедура1
кон
алг Процедура1
нач
    вывод а
кон
```

---

<sup>6</sup> В школьном алгоритмическом языке глобальные переменные объявляются (описываются) до основной части программы, в языке программирования Python описание глобальных переменных проводится по-особому.

<sup>7</sup> В школьном алгоритмическом языке основная часть программы размещается раньше всех вспомогательных процедур и функций.

```

цел а | Глобальная переменная
алг Основная часть программы
нач
    Процедура1
вывод а
кон
алг Процедура1
нач
    а := 200
кон

```

Еще один случай, который следует рассмотреть, – совпадение имен локальной и глобальной переменных. Как, по-вашему, что будет выведено на экран в результате выполнения следующей программы:

```

цел а
алг Основная часть программы
нач
    а := 100
    Процедура1
кон
алг Процедура1
нач цел а
    а := 1
вывод а
кон

```

Правильный ответ – 1 (говорят, что в этом случае локальная переменная «закрывает» собой глобальную).

Предлагаем читателям обратить внимание на эти важные обстоятельства.

В заключение заметим, что процедура может в процессе своей работы использовать любую другую процедуру (это было продемонстрировано выше) или саму себя. В последнем случае такие вспомогательные программы называется «рекурсивными». Рекурсивным процедурам посвящена следующая статья.

## Рекурсия. Рекурсивные функции и процедуры<sup>8</sup>

Схватив себя за эту косичку (своего парика – **авт.**), я изо всех сил дернул вверх и без большого труда вытащил из болота и себя, и своего коня... Если вы думаете, что это легко, попробуйте проделать это сами.

*Барон Мюнхгаузен<sup>9</sup>*

Рекурсией (от латинского *recursio* – возвращение) называют ситуацию в программе, когда процедура или функция вызывает в качестве вспомогательной саму себя. Непонятно? – Давайте разбираться.

Предположим в программе нужно рассчитать значение  $x$ :

$$x = \frac{2 + \sqrt{2}}{5 + \sqrt{5}} + \frac{5 + \sqrt{5}}{13 + \sqrt{13}} + \frac{11 + \sqrt{11}}{8 + \sqrt{8}} + \frac{14 + \sqrt{14}}{7 + \sqrt{7}}$$

<sup>8</sup> Задания на использование рекурсивных процедур и функций используются в ЕГЭ по информатике.

<sup>9</sup> Надеемся, вам известен этот персонаж.

Видно, что в выражении имеются похожие фрагменты – дроби. Желательно для расчета

$$\frac{a + \sqrt{a}}{b + \sqrt{b}}$$

дробей оформить функцию, которая вычисляла бы значения отдельных дробей вида при любых значениях  $a$  и  $b$ , а затем использовать ее для расчетов.

На школьном алгоритмическом языке такая функция имеет вид:

```
алг вещь Др (арг цел a, b)
нач
    знач := (a + sqrt(a)) / (b + sqrt(b)) | Значение функции
кон
```

В ней используется стандартная функция школьного алгоритмического языка `sqrt`, рассчитывающая квадратный корень из своего аргумента.

Так вот, при оформлении собственной функции (или процедуры) в качестве вспомогательной может быть применена и сама эта функция (или процедура), то есть можно использовать рекурсию. Для иллюстрации создадим функцию с рекурсией (такие функции называются «рекурсивными») для расчета факториала натурального числа  $n$ . Такую функцию можно создать, имея в виду, что  $n! = (n - 1)! \times n$ :

```
алг цел Факториал (арг цел n)
нач
    знач := Факториал(n - 1) * n | Рекурсивный вызов функции Факториал
кон
```

На самом деле эта функция оформлена не по правилам (и при выполнении программы появится сообщение об ошибке), но об этом – чуть позже. Здесь же мы обсудим закономерный вопрос, который уже наверное появился у вас – а зачем нужно было «городить огород», когда можно было создать такую, естественную и понятную, функцию:

```
алг цел Факториал (арг цел n)
нач цел произведение, k
    произведение := 1
    нц для k от 2 до n
        произведение := произведение * k
    кц
    знач := произведение
кон
```

На этот ваш вопрос в данном случае трудно ответить убедительно. Но давайте рассмотрим еще одну задачу: «Разработать функцию для расчета  $k$ -го члена последовательности Фибоначчи (последовательность Фибоначчи образуют числа 1, 1, 2, 3, 5, 8, 13, ...)».

Если вы хотите попробовать создать такую функцию самостоятельно (не используя рекурсию), то не смотрите на приведенный далее вариант:

Использованы следующие величины:  
 очередная расчисляемая элемент последовательности;  
 номер очередного элемента; элемент, предшествующий очередному элементу;  
 элемент, предшествующий элементу пред.

```

кон
нач
  очеред := пред
пк
  пред := очеред
  очеред := пред + пред
  | Подумать к расчёту следующего элемента
  | Очередной элемент равен сумме пред и пред
  нн нн нн нн нн нн нн нн нн нн
  1 := пред
  1 := пред
нн нн нн нн нн нн нн нн нн нн
  (к) нн нн нн нн нн нн нн нн нн нн

```

Быстро ли вы получили требуемую функцию? А теперь посмотрите, как просто и логично выглядит рекурсивный вариант функции<sup>10</sup>:

```

алг цел Фиб (арг цел к)
нач
  знач := Фиб(к - 2) + Фиб(к - 1)
кон

```

Такое оформление полностью соответствуют закону построения последовательности Фибоначчи – очередной элемент последовательности равен сумме двух предыдущих. При нем не требуется применять оператор цикла и думать над последовательностью расчета значений пред и предпред.

Теперь о том, почему приведенные рекурсивные функции оформлены неправильно и почему при выполнении программы появится сообщение об ошибке. Дело в том, что, как указывалось в предыдущей статье, при каждом вызове вспомогательной функции (или процедуры) для нее отводится место в оперативной памяти, которое «освобождается» после завершения ее (функции/процедуры) работы. Но если во вспомогательной функции имеется рекурсия, то вызовы вспомогательных функций все продолжаться и продолжаться до тех пор, когда место в памяти будет исчерпано. Чтобы устранить этот недостаток, необходимо так оформлять функции (процедуры), чтобы рекурсивные вызовы осуществлялись по условию, которое когда-то станет ложным. Например, для приведенной функции Факториал это условие записывает так:

```

алг цел Факториал (арг цел n)
нач
  если n > 1
  то
    | Рекурсивный вызов функции Факториал
    знач := Факториал(n - 1) * n
  иначе | Известное значение
    знач := 1
  все
кон

```

или

<sup>10</sup> Он также сознательно оформлен неправильно.

```

алг цел Факториал (арг цел n)
нач
  если n = 1
  то
    знач := 1
  иначе
    знач := Факториал(n - 1) * n
  все
кон

```

а для функции Фиб:

```

алг цел Фиб (арг цел k)
нач
  если k = 1 или k = 2
  то
    | Значение функции известно
    знач := 1
  иначе
    | Рекурсивный вызов функции Факториал
    знач := Фиб(k - 2) + Фиб(k - 1)
  все
кон

```

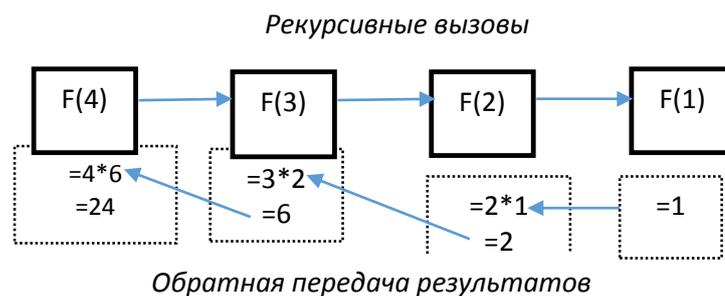
или

```

алг цел Фиб (арг цел k)
нач
  если k >= 3
  то
    | Рекурсивный вызов функции Факториал
    знач := Фиб(k - 2) + Фиб(k - 1)
  иначе
    | Значение функции известно
    знач := 1
  все
кон

```

Пример работы программы при вычислении значения факториала числа 4 показан на схеме и в таблице ниже:



F(n)	F(4) →	F(3) →	F(2) →	F(1)
Значение функции	24 ← 4 × 6	6 ← 3 × 2	2 ← 2 × 1	1
n	4	3	2	1

Ответ: 24.

# Три типовые задачи программирования

## 1. Методы заполнения массивов

Напомним, что в программировании массив – это структура данных, предназначенная для хранения группы значений одного типа. Каждое отдельное значение массива называется «элемент массива». Элементы массива при хранении массива в памяти имеют уникальные порядковые, идущие подряд, номера – «индексы». Чтобы обратиться в программе к значению некоторого элемента массива, надо указать имя массива и в скобках (квадратных или круглых в зависимости от языка программирования) – индекс элемента.

Заполнить массив значениями можно несколькими методами.

Если на момент написания программы известны значения всех элементов массива, то эти значения могут быть записаны в массив с помощью операторов присваивания:

```
м[1] := ...
м[2] := ...
...
```

Иногда известен закон, которому подчиняются значения элементов. Например, значение каждого элемента равно его индексу. Пусть размер массива равен 10. Можем так оформить фрагмент, связанный с заполнением массива:

```
м[1] := 1
м[2] := 2
...
м[10] := 10
```

Обозначив меняющийся при повторении индекс элемента массива так, как это традиционно делается в программировании, – именем  $i$ , можем оформить всё в виде оператора цикла с параметром:

```
нц для i от 1 до 10
  м[i] := i
кц
```

А как заполнить массив так, чтобы он был таким, как, например, следующий:

Значение элемента массива	20	19	18	...	2	1
Индекс $i$	1	2	3	4	19	20

Можно ли установить связь между индексом элементов массива и их значениями в данном случае? – Можно. Значение  $i$ -го элемента равно  $20 - i + 1$  (убедитесь в этом!).

Значит, можем так заполнить нужный массив:

```
нц для i от 1 до 20
  м[i] := 20 - i + 1
кц
```

Опишем также методику заполнения массива известными значениями в ходе выполнения программы. В таких случаях нужно ввести в программу значения всех элементов и записать их в массив. Это также можно сделать с использованием оператора цикла с параметром:

```
нц для i от 1 до 10
  вывод нс, "Введите значение очередного элемента массива "
  ввод м[i]
кц
```

### Задания для самостоятельной работы

1. Заполнить массив двадцатью числами 13.
2. Заполнить массив двадцатью пятью первыми натуральными числами (1, 2, ..., 25).
3. Заполнить массив степенями числа 2 (от  $2^1$  до  $2^{10}$ ). Операцию возведения в степень не использовать.
4. Заполнить массив двадцатью первыми числами Фибоначчи (последовательность чисел Фибоначчи образуются так: первые два ее члена равны 1, каждый следующий, начиная с третьего, равен сумме двух предыдущих – 1, 1, 2, 3, 5, 8, ...).

### 2. Удаление из массива первого элемента со сдвигом всех расположенных справа от него элементов на одну позицию влево

Пример показан на рисунке:

5	12	8	0	14	5	1	3	23	4	8	0	9
---	----	---	---	----	---	---	---	----	---	---	---	---

а) исходный массив

12	8	0	14	5	1	3	23	4	8	0	9	9
----	---	---	----	---	---	---	----	---	---	---	---	---

б) конечный массив

Здесь приведём полезный приём – если сразу оператор цикла с параметром оформить сложно, следует выписать действия, являющиеся телом этого оператора, принять изменяющуюся величину в качестве параметра оператора цикла и определить его начальное и конечное значения.

Операторы, осуществляющие сдвиг необходимых элементов:

```
м[1] := м[2]
м[2] := м[3]
...
м[n - 1] := м[n]
```

можно оформить с использованием оператора цикла с параметром в виде:

1) если в качестве параметра цикла  $i$  принять индекс элемента, на место которого записывается следующий за ним элемент:

```
нц для i от 1 до n - 1
  м[i] := м[i + 1]
кц
```

2) если в качестве параметра цикла  $i$  принять индекс элемента, который записывается на место предыдущего элемента:

```
нц для i от 2 до n
  м[i - 1] := м[i]
кц
```

**Внимание!** После сделанных изменений следует учесть, что количество оставшихся элементов исходного массива уменьшилось на 1 (последний элемент теперь записан в «другом месте» и учитываться не должен).

### Задание для самостоятельной работы

В массиве из 18 элементов записаны фамилии учеников класса. В течение учебного года выбыл ученик, фамилия которого записана в 5-м элементе массива. Получить массив, в котором фамилии выбывшего ученика не будет.

### 3. Сдвиг всех значений элементов на одну позицию вправо и запись нового значения в первый элемент массива.

Пример для нового значения, равного 13, показан на рисунке:

5	12	8	0	14	5							
1	2	...										

а) исходный массив

13	5	12	8	0	14	5						
1	2	...										n

б) конечный массив

Можно ли операторы, осуществляющие сдвиг всех элементов исходного массива, оформить так:

```
м[2] := м[1]
м[3] := м[2]
...
м[n] := м[n - 1]
```

– Нет, не можем! (Почему, установите самостоятельно.) Нужно проводить сдвиг, начиная с конца массива:

```
м[n] := м[n - 1]
м[n - 1] := м[n - 2]
...
м[3] := м[2]
м[2] := м[1]
```

С использованием оператора цикла с параметром соответствующий фрагмент оформляется в виде:

```
нц для i от n до 2 шаг -1
  м[i] := м[i - 1]
кц
```

или

```
нц для i от n - 1 до 1 шаг -1
  м[i + 1] := м[i]
кц
```

После сдвига можно в первый элемент массива записать новое число:

```
м[1] := ...
```

#### Задание для самостоятельной работы

В массиве из 18 элементов записаны фамилии учеников класса. В течение учебного года в класс поступил ученик, фамилия которого должны быть записана в 10-м элементе массива. Получить новый массив, в котором фамилии будут представлена и фамилия этого ученика.

## История информатики

### Мобильный телефон 1984 года



Он стоил больше 4000 долларов.

*Задача, которую вы решаете, может быть очень скромной, но если она бросает вызов вашей любознательности и если вы решите её собственными силами, то вы сможете испытать ведущее к открытию напряжение ума и насладиться радостью победы.*

Джордж Пойя

## Дмитрий и новые коллеги

Программист Дмитрий только что вышел на новую работу. В первый день он познакомился с тремя коллегами из отдела: Игорем, Михаилом и Петром. Один из них блондин, другой — брюнет, а третий — рыжий. Ещё из их путаных рассказов Дмитрий запомнил:

- что Пётр приходится дядей рыжему коллеге и работает в компании дольше племянника,
- брюнет устроился в компанию позже Игоря,
- коллега, который работает дольше всех, не имеет братьев и сестёр.

Дмитрий внимательно слушал новых знакомых, но от волнения информация перепуталась у него в голове и теперь он не может вспомнить, как кого зовут.

Определите имена сотрудников и заодно цвет их волос.

## Оригинальная система счисления

В некоторой непозиционной системе счисления цифры обозначаются геометрическими фигурами. Ниже представлены некоторые числа этой системы счисления и соответствующие им числа десятичной системы счисления:

Неизвестная система	Десятичная система
$\square \times$	9
$\times \square$	11
$\times \square \times$	19
$\circ \times \circ$	190
$\odot \circ \odot$	1900

Определите числовой эквивалент использованных в ней символов.

## Числа-палиндромы

Некоторое десятичное число  $X$ , не превышающее 10, является палиндромом в трёх различных системах счисления (число-палиндром — это число, одинаково читаемое как слева направо, так и справа налево, например,  $707_8$ ,  $2112_4$  и т. п.). Какое это число? В каких системах счисления оно является палиндромом? Однозначное число палиндромом не является.

## Блюдце, стакан и чашка

Стакан и чашка весят больше, чем стакан и блюдце. Что весит больше — чашка или блюдце? Задача предназначена для учащихся начальной школы.

## Девять чисел в таблице

В клетки таблицы  $3 \times 3$  вписаны числа от 1 до 9. Катя нашла сумму чисел в каждом из квадратов  $2 \times 2$ , а затем сложила полученные суммы. Какова наименьшая возможная сумма этих четырёх сумм?

## Где машинка?

В одной из трёх коробок находится игрушка – машина. В какой именно, если из трёх надписей на коробках верной является только одна.



*Машина в этой коробке*



*Машина не в этой коробке*



*Машина не в 1-й коробке*

Это полезно знать

## О високосных годах и старом и новом стиле

Как известно, бывают года, которые называют «високосными». В них не 365 дней, как обычно, а 366 (в календаре добавляется 29 февраля). Почему понадобилось использовать такие года? Дело в том, что вокруг Солнца Земля делает полный оборот не за 365 суток, а за несколько большее время. Когда-то в древности этот период был принят равным  $365\frac{1}{4}$  суток. Получалось, что за четыре года календарь, так сказать, «забегал вперед» по отношению к реальности на один день. Конечно, один день «отклонения» от принятых норм и правил в жизни многих цивилизаций, например, в датах посева и сбора урожая, времени разлива рек и т.п., можно было и не заметить. А если «сдвиг» составит 30 дней? (Это могло произойти через 120 лет после введения календаря.). Такое несоответствие уже могло существенно повлиять на жизнь людей. Поэтому, чтобы компенсировать указанное «забегание», и было решено каждый четвертый год «удлинять» календарь на один день. Календарь, учитывающий такое правило, назывался «юлианским», поскольку он был введен римским императором Юлием Цезарем с 1 января 45 года до н.э.

Но в XVI веке был введен в употребление новый способ определения високосности года.

Необходимость исправления была вызвана тем, что период обращения Земли вокруг Солнца фактически на несколько минут меньше  $365\frac{1}{4}$  суток (если быть точным, то он равен 365 суток 5 часов 48 минут и почти 47 секунд, то есть примерно 365,2422 суток). Из-за этого получалось, что юлианский календарь немного «отставал от жизни», и с течением времени Новый год приближался к весне. Если бы не нововведение, то когда-то в будущем он пришелся бы на время цветения подснежников (не ☺). День весеннего равноденствия во второй половине XVI века по юлианскому календарю должен был наступать 21 марта, а фактически он наступал 11 марта. А ведь от этого дня католическая церковь рассчитывала дату наступления праздника Пасхи! Поэтому в 1582 году Папа Римский Григорий XIII повелел 5 октября считать 15 октября 1582 года (то есть «укоротить» год и, так сказать, «наверстать»

накопившееся к тому времени отставание от реальности в 10 дней), а в будущем последний год любого столетия считать високосным (366 дней) только в том случае, если его номер делится на 400 (например, 1600, 2000)

В таком календаре (его называют «григорианским») время полного оборота Земля вокруг Солнца принимается равной 365,2425 суток. Это число можно записать так:

$$365,2425 = 365 + 0,25 - 0,01 + 0,0025 = 365 + \frac{1}{4} - \frac{1}{100} + \frac{1}{400}$$

Отсюда следует распределение високосных годов, соответствующее этому периоду:

- год, номер которого кратен 400, — високосный;
- остальные года, номер которых кратен 100, — невисокосные;
- остальные года, номер которых кратен 4, — високосные.

Таким образом, 1600-й и 2000-й года были високосными, а 1700-й, 1800-й и 1900-й — високосными не были.

В России григорианский календарь был введен только в январе 1918 года. Проблему решили просто: после 31 января 1918 года следовало сразу 14 февраля (см. ниже). Таким образом отменялась почти двухнедельная разница между юлианским и григорианским календарями.

Поэтому даты многих исторических событий, произошедших до перехода на григорианский календарь, принято называть «датами по старому стилю».



1918 год						
ЯНВАРЬ				ФЕВРАЛЬ		
1	8	15	22	29	18	25
2	9	16	23	30	19	26
3	10	17	24	31	20	27
4	11	18	25		14	21
5	12	19	26		15	22
6	13	20	27		16	23
7	14	21	28		17	24

Различие между старым и новым стилями появляется благодаря тому, что в старом стиле есть даты, отсутствующие в новом стиле (29 февраля 1700 года, 29 февраля 1800 года и 29 февраля 1900 года). В каждый из этих дней (по старому стилю!) число дней, на которое новый стиль опережает старый стиль, увеличивается на 1. В настоящее время это число равно 13. Расхождение между старым и новым стилями приведено в таблице (вы с легкостью продолжите её в любую сторону).

Даты (по новому стилю) включительно	Число дней, на которые новый стиль опережает старый стиль
13 марта 1900 г. — 13 марта 2100 г.	13
12 марта 1800 г. — 12 марта 1900 г.	12
11 марта 1700 г. — 11 марта 1800 г.	11

## Пять вопросов

Ответы на приведенные вопросы найдите в Интернете или по другим источникам информации и пришлите их в редакцию.

1. Кто, нападая на свою жертву, закрывает глаза, чтобы бьющаяся добыча их не поранила?
2. Что имели в виду братья Гонкуры, утверждая, что она при своем появлении никогда не бывает шедевром, а становится им потом?
3. Какой город в Смоленской области издревле славился своими пряниками и коврижками?
4. Кто из чемпионов мира по шахматам умел легко перемножать в уме трехзначные числа?
5. Какая популярная музыкальная группа приехала в Москву из Таганрога?

## «Ломаем» голову

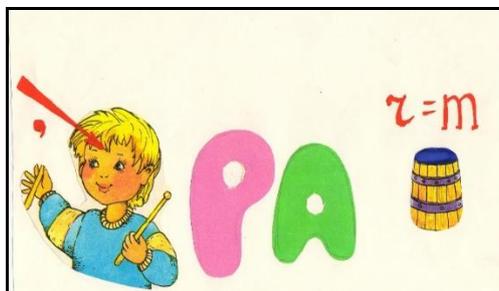
### Ребусы по информатике

Приведенные ниже ребусы несколько лет назад подготовил Бояков Данил, тогда учащийся лицея «Технический», г. Самара, учитель **Боякова И.А.** Решите их и дайте определение каждому из найденных терминов.

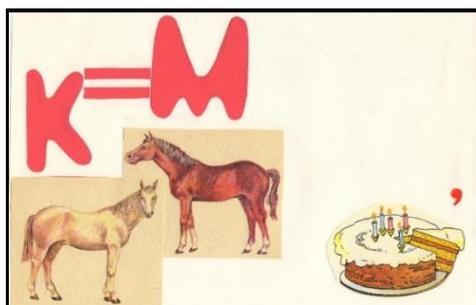
#### Ребус № 1



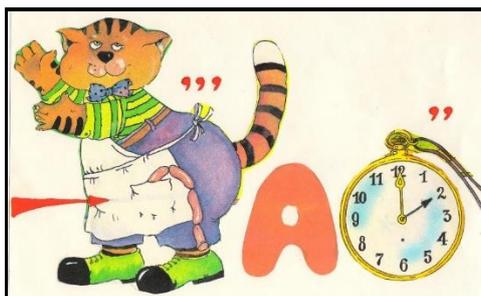
#### Ребус № 2



### Ребус № 3



### Ребус № 4

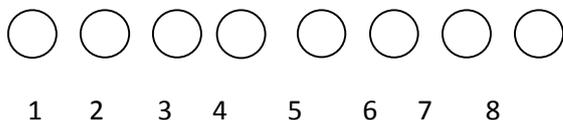


## Лишний рубль

Когда-то давно, когда цены были другие, студент Митя собрался ехать домой к родителям, и ему надо было на билет примерно 50 рублей. Он занял у двух своих друзей по общежитию по 25 рублей у каждого и поехал за билетами. Они ему обошлись в 45 рублей. Когда он шел назад, то встретил третьего знакомого, которому срочно надо было три рубля. Митя их ему одолжил. Когда он пришел в общежитие, то отдал первым двум своим друзьям по рублю. Приехав к родителям, Митя взял у них 48 рублей, чтобы отдать долг друзьям ( $24 + 24 = 48$ ). Когда студент вернулся обратно, то встретил третьего своего друга, который вернул ему 3 рубля. В результате у Мити на руках оказалось  $48 + 3 = 51$  рубль. Но ведь занимал он всего 50 рублей? Откуда взялся лишний рубль?

## Восемь монет

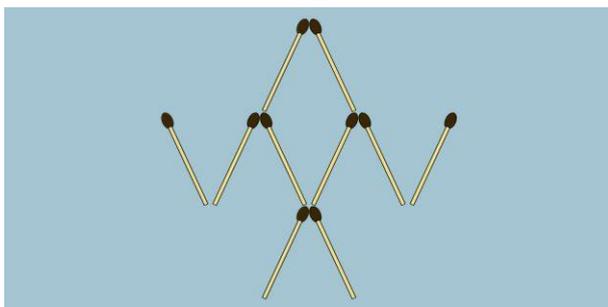
Восемь монет выложены в ряд:



Надо собрать их в две группы по 4 штуки в каждой группе (в которой монеты, например, лежат одна на другой), перекладывая за каждый ход одну монету. При этом перекладываемую монету можно соединять лишь с монетой (или монетами), отделенной от нее двумя монетами (разрозненными или объединенными в группу).

## Переставить три спички

Переставьте три спички так, чтобы рак полз вниз, а не вверх.



## Коля, Оля и тетя Поля

Когда Коля был молод, как Оля,  
Столько лет было тетушке Поле,  
Сколько Коле теперь вместе с Олей.

Вопрос:

*Сколько лет было Коле,  
Когда тетушка Поля  
Была в возрасте Коли?*

Ответ, пожалуйста, присылайте в редакцию.

## Десять монет и клоп

Есть 10 монет. Шесть из них весом ровно 10 г и четыре — ровно 11 г. Монеты выглядят совершенно одинаково. Имеются очень точные двухчашечные весы, но, к сожалению, на результат взвешивания влияет вредный невидимый клоп, который при каждом взвешивании садится на чашу весов, но на какую именно — неизвестно. Вес клопа ровно 1 г. (Клоп не садится на весы, если на них ничего не взвешивается.)

За какое минимальное число взвешиваний без гирь можно обнаружить вредное насекомое, если монеты нельзя взвешивать повторно?

## Какая буква пропущена?

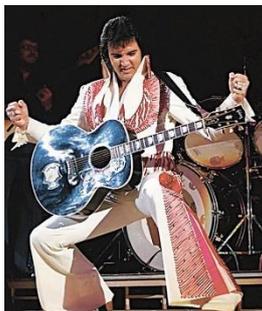
Приведенная ниже последовательность букв подчиняется некоторой чёткой закономерности.

П В С  П С В

Определите, какой может быть эта последовательность, и выясните, какую пропущенную букву нужно вставить вместо квадратика.

## Числовой ребус с Элвисом Пресли

Решите, пожалуйста, числовой ребус, в котором, как обычно в таких головоломках, одинаковыми буквами зашифрованы одинаковые цифры, разными буквами – разные цифры.



$$\begin{array}{r} \text{ЭЛВИС} \\ + \text{ЭЛВИС} \\ \hline \text{ПРЕСЛИ} \end{array}$$

*Указания по выполнению*

Определите, какой может быть цифра С, и исследуйте все возможные варианты.

## Робинзона Крузо и таблетки

Однажды на необитаемом острове Робинзона Крузо укусила очень ядовитая змея.

Случайно, у Робинзона были с собой четыре таблетки: две таблетки лечебные, а две – для усиления лечебного эффекта. Причём эти таблетки необходимо было принимать только попарно: одну лечебную вместе с дополнительной, которая сама по себе не спасает от яда.

Первый раз надо принять одну лечебную таблетку и одну – дополнительную к ней. Также надо поступить и на следующий день.

Но, к несчастью, все четыре таблетки оказались в одной баночке, и проблема в том, что они по внешнему виду совершенно не отличаются друг от друга. Теперь непонятно, где лечебные, а где – дополнительные к ним.

Как может поступить Робинзон Крузо, чтобы спастись?

*Задачу предложил Л.А. Штейнгарц, Израиль*

Это полезно знать

## Советы от фирмы Naked Science

Тем, кто решил стать программистом, но не выбрал, в какой сфере будет работать, стоит обратить внимание на Python — это универсальный язык программирования, который подходит для решения разнообразных задач и различных платформ, от мобильных до серверных.

Если планируете заняться созданием мобильных приложений, присмотритесь к Java. Программисты, занимающиеся созданием сайтов, делятся на frontend-разработчиков, отвечающих за визуальную составляющую сайта, и backend-разработчиков, которые программируют и поддерживают серверную часть веб-приложений.

Если же нравится скорее не создавать, а даже ломать, то вам путь в **QA-тестировщики**. Специалист по тестированию занимается проверкой программ и приложений, ищет программные ошибки и недоработки. Образно говоря, если программист-разработчик пытается что-то создать, то задача тестировщика — сломать это.

Тест от фирмы Naked Science на тему «Можете ли вы стать программистом?» приведен ниже.

## Конкурс № 51

Задание этого конкурса связано с поиском информации.

Ответьте, пожалуйста, на вопрос: «Какие математические инструменты изображены на картине с выдающимся русским учёным Михаилом Васильевичем Ломоносовым, сидящим за столом?» Расскажите об этой картине найденную информацию.

## Конкурс № 52 «Тест для тех, кто хочет стать программистом»

Это тест на знание базовых понятий в сфере программирования. Что-то можно вспомнить из курса школьных предметов, а на некоторые вопросы помогут ответить эрудиция, интуиция и логическое мышление.

1. Найдите общее между всеми этими предметами и определите, что здесь лишнее?

- А. Торшер
- Б. Луна
- В. Спичка
- Г. Костер

2. Это обычные дроби. Какая должна чувствовать себя чужой в окружении других?

- А.  $\frac{3}{4}$
- Б.  $\frac{3}{6}$
- В.  $\frac{3}{8}$
- Г.  $\frac{3}{10}$

3. Как правило, первая программа, которую пишет каждый программист, начиная изучать новый язык программирования по книгам, — это программа, выводящая на экран текст...

- А. Game started!
- Б. You did it!
- В. Hello, world!
- Г. Just Another Programmer Born!

4. Продолжите фразу: «Единственный язык, напрямую выполняемый ЭВМ, — это ...»

- А. ... Python
- Б. ... исходный код
- В. ... машинный язык
- Г. ... Си

5. Продолжите цепочку: 1000011000111...

- А. ...01100...
- Б. ...00111...
- В. ...11100...

6. Вы уже определились, что хотите стать веб-разработчиком и делать сайты (см. рубрику «Это полезно знать»)? Если вы решили стать Frontend-разработчиком, то тогда вам нужно освоить...

- А. Java, Objective-C, Swift
- Б. HTML, CSS, JavaScript
- В. C++, PHP, Ruby
- Г. Английский язык, математику, метод слепой печати на клавиатуре

7. Когда-то в школе на уроках информатики все изучали язык программирования Basic. Узнайте его по фрагменту кода. Известны ли вам остальные представленные ниже языки?

- А. cout << "Привет!"
- Б. System.out.println("!")
- В. PRINT "Привет"
- Г. println("Привет!")

8. Предположим, птицы с рождения умеют программировать. Некоторые птицы — жаворонки. Все жаворонки обязательно знают три языка программирования. Что из этого верно?

- А. Все птицы, которые знают только два языка программирования, не являются жаворонками
- Б. Все птицы, которые знают три языка программирования, являются жаворонками
- В. Птицы, которые знают больше трех языков, не являются жаворонками
- Г. Совы не знают языков программирования



9. Некоторые программисты любят математику. Некоторые любители математики любят слушать Баха. Из этого утверждения следует, что ...

- А. ... среди программистов обязательно должны быть любители музыки Баха
- Б. ... все программисты слушают Баха
- В. ... любители Баха есть и среди программистов, и среди математиков
- Г. Здесь нет правильного ответа

Ответы присылайте в редакцию (в ответах на конкурс № 52 можно отвечать не на все вопросы).

Укажите в письме свои фамилию и имя, населенный пункт, номер школы и фамилию, имя и отчество учителя информатики. Срок представления ответов — 31 октября 2022 г.

Книжная полка

## Полезная книга

Издательство ДМК-Пресс предлагает книгу «Подготовка к успешной сдаче ОГЭ по информатике» (авторы Д.М.Златопольский, В.А.Мишин)

<https://dmkpress.com/catalog/computer/handbooks/978-5-93700-121-4/>

Большинство книг, связанных с Основным государственным экзаменом по информатике, представляют собой сборники заданий для самопроверки или «решебники», в которых приводятся решение заданий без подробных комментариев. Указанная книга отличается от них тем, что в ней рассказывается о типовых вариантах заданий ОГЭ и особенностях каждого из них, приводятся различные методы их выполнения, даются полезные советы и рекомендации, обеспечивающие ускорение выполнения заданий и уменьшающие вероятность ошибки. После знакомства с этими вопросами читатель может проверить свои знания и умения, выполнив предлагаемые в книге задания для самостоятельной работы.

Кроме учащихся 9-го класса, готовящихся к сдаче ОГЭ, книгу могут использовать учителя и преподаватели информатики. Она будет также полезна учащимся 10-го и 11-го классов, поскольку многие вопросы, рассмотренные в книге, являются полезными для подготовки к ЕГЭ по информатике.

Приобрести книгу по цене, гораздо меньшей, чем в магазинах (даже с учетом стоимости доставки), можно через сайт издательства.

## Что такое «десятичная система счисления»? (о двух терминах информатики)

Приведем ответ на вопрос, указанный в названии статьи, согласно книге [1] – это традиционная система счисления с основанием 10 (о понятиях «традиционная система счисления» и «основание» – см. [1]). В тоже время в авторитетном издании [7, с. 11] «десятичной» называется система счисления, применявшаяся в Египте в начале второго тысячелетия до н.э., хотя в этой системе использовались не 10 знаков, а 7 [8] (рис. 1):

Знак	1	10	100	1000	10000	100000	1000000
Значение	1	10	100	1000	10000	100000	1000000

Рис. 1. Знаки, использовавшиеся в древнеегипетской системе счисления

Числа, не являющиеся степенью 10, записывались путем повторения этих цифр. Каждая цифра могла повторяться от 1 до 9 раз. Например, число 4622 обозначалось следующим образом (см. рис. 2):



Рис. 2. Пример записи числа в древнеегипетской системе счисления

Анализ показывает, что в египетской системе число представлялось в виде знаков, соответствующих степеням числа 10. Именно это обстоятельство, очевидно, дало основание авторам [7] называть такую систему счисления «десятичной», правда, с уточнением «непозиционная»<sup>11</sup>.

Еще один пример – способ записи чисел в Древнем Вавилоне. В вавилонских текстах, написанных характерной «клинописью», использовались два всего два знака: «стоячий клин» (𐀀) для обозначения единиц и «лежачий клин» (𐀁) для обозначения десятков. Примеры для чисел 1–59 приведены на рис. 3.

<sup>11</sup> Так же названа древнеегипетская система в [2].

𐎶 1	𐎶𐎶 11	𐎶𐎶𐎶 21	𐎶𐎶𐎶𐎶 31	𐎶𐎶𐎶𐎶𐎶 41	𐎶𐎶𐎶𐎶𐎶𐎶 51
𐎷 2	𐎷𐎶 12	𐎷𐎶𐎶 22	𐎷𐎶𐎶𐎶 32	𐎷𐎶𐎶𐎶𐎶 42	𐎷𐎶𐎶𐎶𐎶𐎶 52
𐎸 3	𐎸𐎶 13	𐎸𐎶𐎶 23	𐎸𐎶𐎶𐎶 33	𐎸𐎶𐎶𐎶𐎶 43	𐎸𐎶𐎶𐎶𐎶𐎶 53
𐎹 4	𐎹𐎶 14	𐎹𐎶𐎶 24	𐎹𐎶𐎶𐎶 34	𐎹𐎶𐎶𐎶𐎶 44	𐎹𐎶𐎶𐎶𐎶𐎶 54
𐎺 5	𐎺𐎶 15	𐎺𐎶𐎶 25	𐎺𐎶𐎶𐎶 35	𐎺𐎶𐎶𐎶𐎶 45	𐎺𐎶𐎶𐎶𐎶𐎶 55
𐎻 6	𐎻𐎶 16	𐎻𐎶𐎶 26	𐎻𐎶𐎶𐎶 36	𐎻𐎶𐎶𐎶𐎶 46	𐎻𐎶𐎶𐎶𐎶𐎶 56
𐎼 7	𐎼𐎶 17	𐎼𐎶𐎶 27	𐎼𐎶𐎶𐎶 37	𐎼𐎶𐎶𐎶𐎶 47	𐎼𐎶𐎶𐎶𐎶𐎶 57
𐎽 8	𐎽𐎶 18	𐎽𐎶𐎶 28	𐎽𐎶𐎶𐎶 38	𐎽𐎶𐎶𐎶𐎶 48	𐎽𐎶𐎶𐎶𐎶𐎶 58
𐎾 9	𐎾𐎶 19	𐎾𐎶𐎶 29	𐎾𐎶𐎶𐎶 39	𐎾𐎶𐎶𐎶𐎶 49	𐎾𐎶𐎶𐎶𐎶𐎶 59
𐎿 10	𐎿 20	𐎿 30	𐎿 40	𐎿 50	

Рис. 3. Запись чисел 1 – 59 в вавилонской системе счисления

Числа, большие 59, записывались в двух и более разрядах. Значение, указанное во втором слева разряде в виде, аналогичном представленному на рис. 3, имело весомость 60, в третьем – весомость 3600 и т. д. Например, число 129 имело вид:



Такой способ записи чисел, как известно, называется «смешанной системой счисления» (подробно о ней – ниже). Авторы [7, с. 11] используют название «смешанная десятично-шестидесятеричная система». Почему «шестидесятеричная» – понятно. А почему «десятично»? Ведь внутри шестидесятеричного разряда используются не 10 «цифр», а две. Причина та же – эти знаки соответствуют степеням числа 10.

По той же причине систему счисления индейцев майя [10] можно назвать «пятерично-двадцатеричной» (хотя внутри двадцатеричного разряда для записи значений применялись три

«цифры» – •, — и ):

0	1	2	3	4
	•	••	•••	••••
5	6	7	8	9
—	•	••	•••	••••
10	11	12	13	14
— —	•	••	•••	••••
15	16	17	18	19
— — —	•	••	•••	••••

Рис. 4. Запись чисел 0–19 в системе счисления майя

### Выводы

Итак, можем сделать вывод, что название «десятичная система счисления» можно использовать в двух случаях:

- 1) когда для записи чисел используются 10 символов (цифр); такой вариант будет носить позиционный характер [2, 4–7, 9];
- 2) когда число представляется в виде знаков, соответствующих степеням числа 10 (это будет непозиционная десятичная система).

Указанное обстоятельство требует уточнения также такого понятия информатики, как «смешанная система счисления». В книге [1] о таких системах говорится (приводим нестрогое определение), что это способ записи чисел, заданных в системе счисления с основанием  $Q$ , с помощью цифр другой  $P$ -ичной системы. В качестве примера можно привести одну из разновидностей двоично-десятичной системы, в которой каждая цифра десятичного числа заменяется соответствующим четырехразрядным двоичным числом<sup>12</sup>.

Например, десятичное число 19 205 463 в двоично-десятичном коде имеет вид:

$$\begin{array}{cccccccc} 0001 & 1001 & 0010 & 0000 & 0101 & 0100 & 0110 & 0011 \\ 1 & | & 9 & | & 2 & | & 0 & | & 5 & | & 4 & | & 6 & | & 3 \end{array}$$

То есть в [1] под понятием «смешанная система счисления» имеется в виду «комбинация» двух позиционных систем счисления. Но выше в качестве смешанной системы счисления упоминались и сочетания, например, непозиционной десятичной системы и позиционной шестидесятеричной, непозиционной десятичной системы и позиционной двадцатеричной.

Так что можно говорить о двух типах смешанных систем счисления:

- 1) «смешанные» с точки зрения позиционности отдельных систем;
- 2) «смешанные» с точки зрения основания отдельных систем.

В заключение – еще один момент, связанный со смешанными системами счисления.

Во многих источниках, связанных с школьной информатикой (учительских сайтах, форумах и т. п.), а также в статье [8] на популярном сайте «Википедия» (в качестве примера такой системы приводится представление времени в виде количества суток, часов, минут и секунд. При этом говорится, что величине « $d$  дней,  $h$  часов,  $m$  минут,  $s$  секунд» соответствует значение:

$$d \times 24 \times 3600 + h \times 3600 + m \times 60 + s \text{ секунд.}$$

Считаем такой пример ошибочным, во-первых, потому, что приведен пример не системы счисления (под которой понимается способ записи/обозначения чисел [1]), а представления значения некоторой физической величины, и, во-вторых, потому, что весомости «разрядов» секунд, минут и часов в примере составляют, соответственно, 1, 60 и 3600, а «разряда» дней –  $24 \times 3600$ , что не соответствует свойствам позиционных систем.

#### Список использованных источников

1. Андреева Е. В., Босова Л. Л., Фалина И. Н. Математические основы информатики. – М.: Бином. Лаборатория знаний, 2005. 328 с.
2. Босова Л. Л., Босова Ю. Л. Информатика. 8 класс. – М.: Бином. Лаборатория знаний, 2014. 155 с.
3. Златопольский Д. М. Системы счисления: Учебные и занимательные материалы. – М.: ЛЕНАНД, 2015. 512 с.
4. И. Г. Семакин, Е. К. Хеннер, Т. Ю. Шеина. Информатика. 10 класс. Базовый уровень: учебник. – М.: БИНОМ. Лаборатория знаний, 2016. — 264 с.
5. К.Ю. Поляков, Е.А. Еремин. Информатика 10 класс (базовый и углубленный уровни). В двух частях. Учебник. Часть 1. – М.: Бином. Лаборатория знаний. 2019. 352 с.
6. Н.Д. Угринович. Информатика 10 класс. Базовый уровень: учебник. – М.: Бином. Лаборатория знаний. 2017. 288 с.
7. Математический энциклопедический словарь. – М.: Советская энциклопедия, 1988. 847 с.
8. Системы счисления. [https://ru.wikipedia.org/wiki/Системы\\_счисления](https://ru.wikipedia.org/wiki/Системы_счисления)
9. Советский энциклопедический словарь. – М.: «Большая советская энциклопедия». – М.: 2012. 1632 с.
10. Цифры маяя. [https://ru.wikipedia.org/wiki/Цифры\\_маяя](https://ru.wikipedia.org/wiki/Цифры_маяя)

<sup>12</sup> Основным преимуществом этого кода, который применяется в калькуляторах, счётчиках, устройствах ввода-вывода, цифровых приборах и т. п. является быстрый перевод в него из десятичной системы и обратно [3].