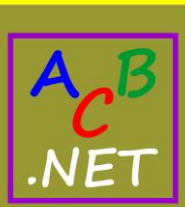
Рубанцев Валерий

Развивающее программирование

Решение задач на языке паскаль



PascalABC.NET

От автора

При слове «паскаль» многие вспоминают убитый временем *Turbo Pascal*, который до сих пор изучают в школе, отбивая у школьников всякий интерес к программированию. Есть ещё много других, тоже древних *паскалей*, но в этой книге мы будем использовать великолепный, мощный, современный *паскаль*, который называется PascalABC.NET. Название грузное, поскольку синтетическое. В нём есть и *паскаль*, и *ABC*, и *.NET*. С *паскалем* понятно – новый *паскаль* наследует всё лучшее, что было в *паскале* его создателя Никлауса Вирта. Буквы *ABC* указывают на назначение *паскаля* – это язык для обучения программированию. И наконец, последние буквы подчёркивают связь с платформой *.NET*. Основной язык этой платформы - *Си-шарп*, у которого PascalABC.NET взял многое и лучшее. Тут следует добавить, что новый *паскаль* немало интересного и полезного позаимствовал и у *Питона*. В итоге PascalABC.NET превратился в идеальное средство обучения и в чрезвычайно удобный инструмент для решения разнообразных задач.

Как и в любом другом деле, при изучении *паскаля* нужна систематическая тренировка, под которой можно понимать решение задач на основные элементы языка программирования. Это могут быть и формальные упражнения, и занимательные задачи. Принципиальной разницы между ними нет, за исключением того, что занимательные задачи имеют сюжет, а потому более интересны.

В этой книге собрано несколько десятков занимательных математических задач всех времён и народов, начиная от самых древних задач, записанных на папирусах и глиняных табличках, до конкурсных задач из журнала *Квантик*.

Решая задачи, вы укрепите умения и навыки в применении таких элементов языка *PascalABC.NET,* как:

- Числовые типы данных – *integer и double*
- Логический тип *boolean*
- Логические операции *or* и *and*
- Арифметические операции
- Простейшие математические функции
- Переменные и константы, внутриблочные переменные

- Операторы объявления/определения и присваивания
- Комбинированные операторы присваивания
- Логические операторы и выражения
- Логические выражения и условные логические операторы *if, if – else*
- Циклы *for, while, repeat – until, foreach*
- Вложенные циклы
- *Массивы, списки, множества*
- Процедуры и функции – без параметров и с параметрами
- Операторы *exit, continue, break*
- Операции ввода и вывода *Read(ln), Write(ln).*

Хотите получить удовольствие и пользу от программирования? – Тогда изучайте *PascalABC.NET*!

Книга адресуется: школьникам, изучающим *PascalABC.NET* на уроках или самостоятельно, учителям информатики и математики, любителям программирования и математики.

Валерий Рубанцев

Условные обозначения, принятые в книге:

Дополнение или замечание

Требование или указание

Исходный код

Задание для самостоятельного решения

Заголовок проекта:

Проект

Исходные коды всех проектов находятся в папке **_PasProjects**

Оглавление

СТАРИННЫЕ ЗАДАЧИ

История математики насчитывает более 6 тысяч лет! Конечно, занимательные задачи появились не вдруг и не сразу, ведь нашим далёким предкам ещё нужно было научиться считать на пальцах и придумать цифры для записи чисел.

Но самым древним «рукописям» - глиняным табличкам и папирусным свиткам – больше 4,5 тысяч лет, а мы находим на них не только «учебные» арифметические задачи, в которых требуется преобразовать какое-либо выражение или вычислить его значение, но и вполне занимательные задачи, не имеющие прямого практического значения.

Главное отличие занимательных задач от учебных заключается в том, что никто не заставляет их решать. Они настолько интересны сами по себе, что их непременно хочется решить.

В этой главе собраны древние и старинные занимательные задачи, и я надеюсь, что они заинтересуют вас точно так же, как и любителей математики много столетий и тысячелетий тому назад.

Египетские кошки

Задача из книги Математический фольклор (Д1):

Каждый из 7 человек имеет 7 кошек. Каждая кошка съедает по 7 мышек. Каждая мышка за одно лето может уничтожить 7 ячменных колосков, а из зёрен одного колоска может вырасти 7 горстей ячменного зерна.

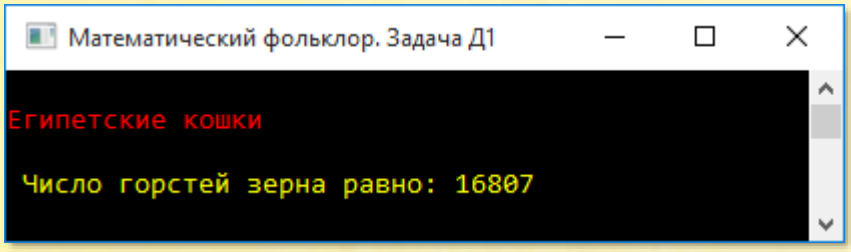
Сколько горстей зерна ежегодно спасается благодаря кошкам?

Эта популярная в Древнем Египте задача сейчас нам кажется очень простой, ведь чтобы решить её, достаточно найти произведение пяти семёрок:

```
uses
   System;

// Математический фольклор. Задача Д1


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
    Console.ForegroundColor := ConsoleColor.Yellow;

    Console.WriteLine(' Число горстей зерна равно: ' +
                      7 * 7 * 7 * 7 * 7);
    Console.WriteLine();
end;


begin
   // заголовок окна:
   Console.Title := 'Математический фольклор. Задача Д1';
   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Египетские кошки');
   Console.ForegroundColor := ConsoleColor.Green;
```

```
    Console.WriteLine();

    Solve();

    Console.WriteLine();
    Console.ForegroundColor := ConsoleColor.Red;
end.
```

Запускаем программу и получаем **ответ**:

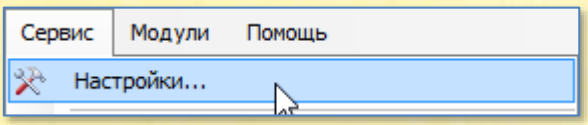


В переводе с древнеегипетских горстей на современные килограммы это составляет примерно **1350**.

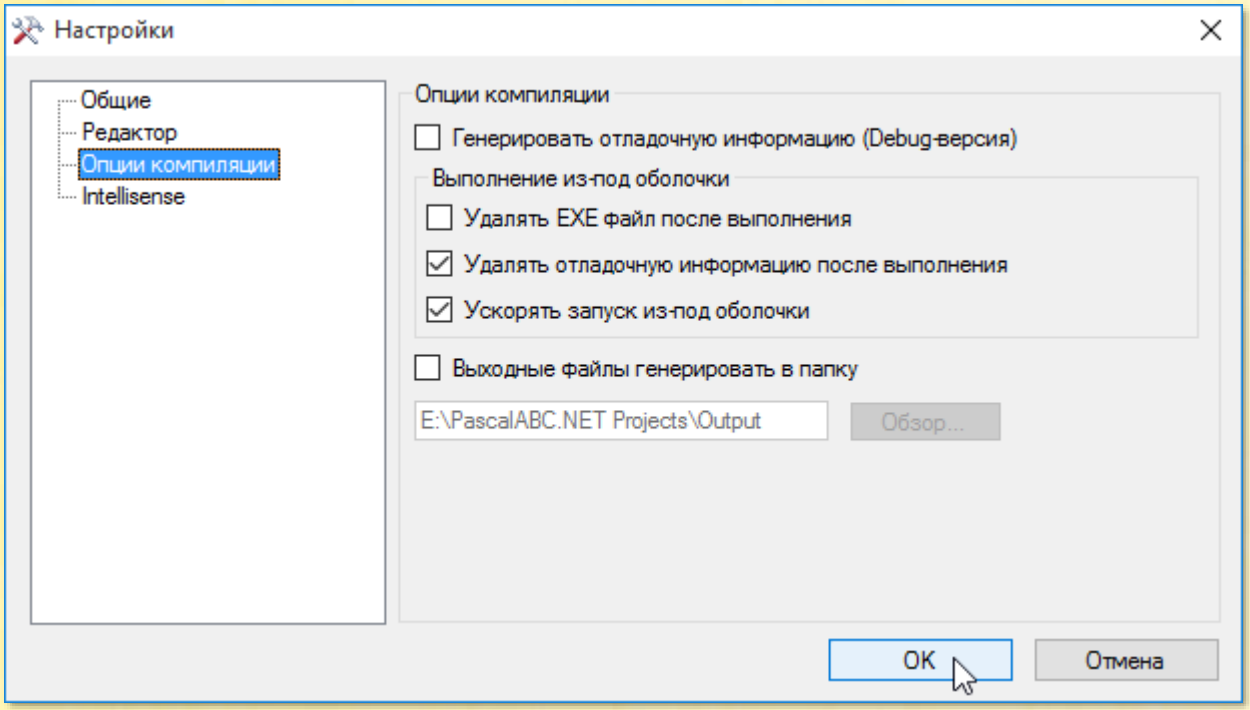
Для древних египтян эта задача была, несомненно, более трудной, чем для нас. Но зато благодаря ей мы узнали, за что древние египтяне так любили кошек!

По умолчанию **исполняемый файл** программы уничтожается после завершения программы. Если вы хотите поделиться программой с друзьями, которые не программируют на *паскале*, то, конечно, лучше передать им не исходный код, а выполняемый файл.

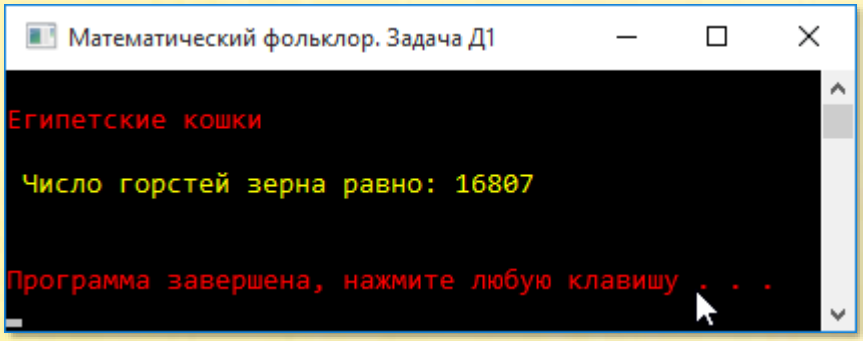
Выполните команду меню **Сервис → Настройки...**:



В открывшемся диалоговом окне откройте вкладку **Опции компиляции**:



Снимите галочку с пункта **Удалять EXE файл после выполнения**. Нажмите кнопку **ОК**. Запустите программу. В папке **Египетские кошки появится** исполняемый файл программы **Египетские кошки.exe**. Щёлкните по его названию. *Консольное окно* появится на экране и тут же исчезнет. Всё дело в том, что при запуске программы из среды разработки окно закрывается только после нажатия на любую клавишу, о чём сообщает надпись в *Консольном окне*:



В скомпилированной программе окно закрывается сразу после того, как весь код выполнен. Чтобы **окно оставалось на экране** до нажатия клавиши ВВОД (ENTER), добавьте перед строкой *end.* оператор:

```
Console.ReadKey();
```

Длинные вызовы методов Console.Write и Console.WriteLine вы можете заменить более короткими вызовами функций Write и Writeln. Тогда код нашей программы станет более коротким и понятным:

```
// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
    Console.ForegroundColor := ConsoleColor.Yellow;

    writeln(' Число горстей зерна равно: ' +
                    7 * 7 * 7 * 7 * 7 * 7);
    Writeln;
end;


begin
   //заголовок окна:
   Console.Title := 'Математический фольклор. Задача Д1';
   Writeln;
   Console.ForegroundColor := ConsoleColor.Red;
   Writeln('Египетские кошки');
   Console.ForegroundColor := ConsoleColor.Green;
   Writeln;

   Solve();

   Writeln;
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

Если вы просто хотите решить задачу, то украшения *Консольного окна* можно убрать, и исходный код ещё более сократится:

```
// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
    writeln(' Число горстей зерна равно: ' +
                    7 * 7 * 7 * 7 * 7 * 7);
```

```
    Writeln;
end;


begin
   Writeln;
   Writeln('Египетские кошки');
   Writeln;

   Solve();

   Writeln;
end.
```

Если процедура (или функция) очень короткая, то для её описания можно пользоваться упрощённым синтаксисом:

```
// РЕШАЕМ ЗАДАЧУ
procedure Solve := writeln(' Число горстей зерна равно: ' +
                           7 * 7 * 7 * 7 * 7);
```

И наконец, короткие действия можно записать непосредственно в главном блоке программы.

Всегла полезно решить задачу несколькими способами.

Найти произведение пяти семёрок несложно. Но что делать, если умножений было бы больше. Тогда лучше воспользоваться циклом *for* и вычислить нужное произведение в функции Solve:

```
// РЕШАЕМ ЗАДАЧУ
function Solve(num: integer; krat: integer): int64;
begin
   Result := 1;
   for var i := 1 to krat do
      Result *= num;
end;
```

Функция **получает** заданное число *num* и количество таких чисел в произведении, а **возвращает** произведение, которое мы печатаем в главном блоке программы:

```
begin
   Writeln;
   Writeln('Египетские кошки');
   Writeln;

   var num := Solve(7, 5);
   writeln(' Число горстей зерна равно: ' + num);
   Writeln;
end.
```

Функция **Solve** универсальна. Она позволяет находить произведение любого целого числа *num* заданное число раз *krat*.

В современной математике произведение пяти семёрок записывают сокращённо: 7^5. Читать это выражение следует так*: семь в пятой степени*. Семёрка – это **основание** степени, пятёрка – **показатель**, а всё выражение в целом - **степень**. Для вычисления степеней в *паскале* имеется функция **Power**, которой нужно передать основание и показатель степени:

Power(основание, показатель) → степень

Эта функция возвращает результат типа *real*. Для преобразования его к типу *int64* нужно применить функцию **Trunc**.

Все вычисления можно провести непосредственно в главном блоке программы, но для общности мы выполним их в функции **Solve**:

```
// РЕШАЕМ ЗАДАЧУ
function Solve(num: integer; krat: integer): int64;
begin
   Result := Trunc(Power(num, krat));
end;
```

Шахматное число

Вам, вероятно, известна легенда про изобретателя шахмат, которого правитель Индии, царь Шерам решил отблагодарить за интересную игру и предложил ему самому назначить цену.

И легенда, и задача встречаются во многих книгах по занимательной математике. Например, вы можете прочитать о ней в книге В. Литцмана *Великаны и карлики в мире чисел*, на странице 26.

Изобретатель - его звали Сета - попросил заплатить ему зерном. А зёрнышки посчитать так: *на первую клетку шахматной доски положить 1 зерно пшеницы, на вторую 2, на третью 4, на четвёртую 8, и так далее, каждый раз увеличивая число зёрен в 2 раза.*

На первый взгляд кажется, что в итоге получится всего горстка пшеницы. Так и подумал царь Шерам и приказал выдать скромному изобретателю гораздо больше – целый мешок пшеницы. Но тот попросил отмерить ровно столько зёрен, сколько получится в результате вычислений.

Шерам не стал спорить с изобретателем и повелел точно вычислить, сколько тому причитается зёрен. Когда придворные математики закончили свои долгие вычисления, то оказалось, что зёрен должно быть так много, что их просто не могло быть ни у Шерама, ни у кого другого на земле.

Если отбросить «мелкие подробности», то нам нужно найти сумму чисел:

$2^0 + 2^1 + 2^2 + 2^3 + 2^4 + \ldots + 2^{63}$

Эти числа образуют **геометрическую прогрессию**, сумма которой равна $2^{64} - 1$.

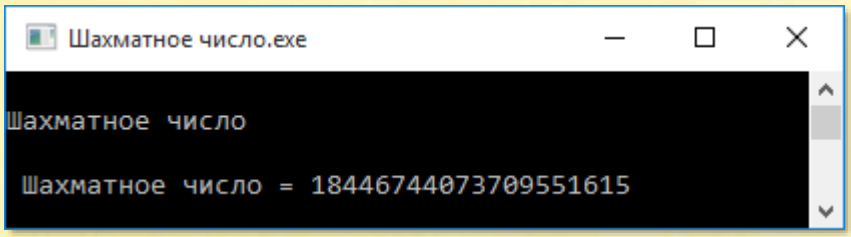
При помощи операции возведения в степень мы без труда решим эту знаменитую шахматную задачу. На бумаге это трудное и долгое занятие, а вот написать программу на *паскале* можно за несколько минут:

```
uses
   System;


// РЕШАЕМ ЗАДАЧУ
function Solve(num: integer; krat: integer): BigInteger;
begin
   Result := TruncBigInteger(Power(num, krat));
end;


begin
   Writeln;
   Writeln('Шахматное число');
   Writeln;

   var num := Solve(2, 64);
   writeln(' Шахматное число = ' + (num-1).ToString());
   Writeln;
```

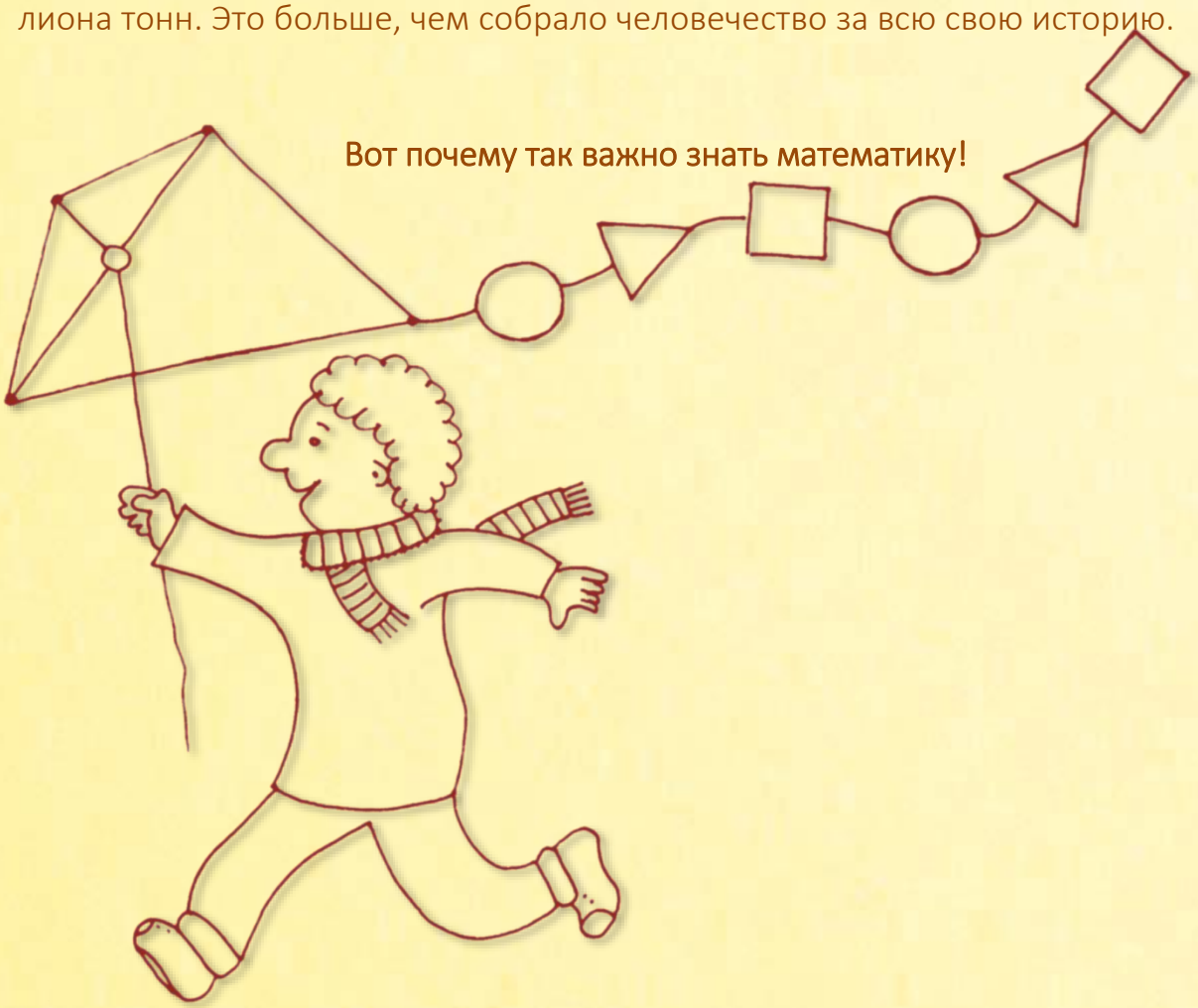
```
end.
```

Степень мы легко вычислим с помощью функции **Power**, но нужно учесть, что для хранения такого длинного числа потребуется переменная типа **BigInteger**.

А число такое - **18 446 744 073 709 551 615:**

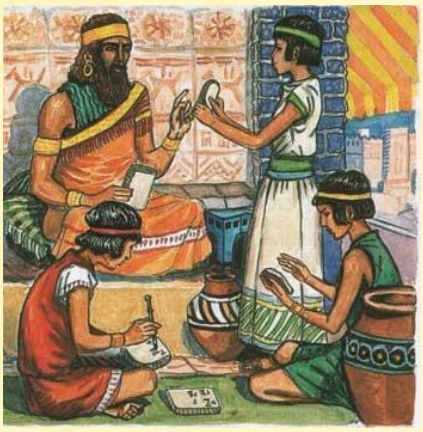


Но число хоть и большое, но непонятное. Давайте переведём зёрна в тонны. Если принять, что зёрнышко весит 0,065 грамма, то общий вес составит около 1,2 трил-лиона тонн. Это больше, чем собрало человечество за всю свою историю.



Вавилонские ладони

Эту задачу придумали в Древнем Вавилоне примерно за 2 тысячи лет до нашей эры:



Длина и ¼ ширины вместе составляют 7 ладоней, а длина и ширина вместе – 10 ладоней.

Сколько ладоней составляют длина и ширина в отдельности?

Принято считать, что ширина не больше длины. Также из условия задачи следует, что ширина в ладонях кратна 4.

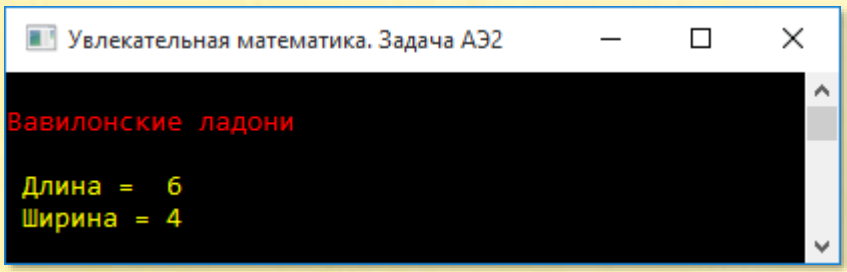
Эти два условия можно без труда перевести на *паскаль*:

```
uses
   System;

// Увлекательная математика. Задача АЭ2


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   for var length := 1 to integer.MaxValue do
      for var w := 0 to length div 4 do
      begin
         var width := w * 4;
         if ((length + width div 4 = 7) and
             (length + width = 10)) then
         begin
            Console.WriteLine(' Длина =  ' + length);
            Console.WriteLine(' Ширина = ' + width);
            Console.WriteLine();
```

```
            exit;
          end
        end;
    Console.WriteLine();
end;


begin
    // заголовок окна:
    Console.Title := 'Увлекательная математика. Задача АЭ2';
    Console.WriteLine('');
    Console.ForegroundColor := ConsoleColor.Red;
    Console.WriteLine('Вавилонские ладони');
    Console.ForegroundColor := ConsoleColor.Green;
    Console.WriteLine();

    Solve();

    Console.WriteLine();
    Console.ForegroundColor := ConsoleColor.Red;
end.
```

Так как мы не знаем, чему равна длина, то перебираем значения в «бесконечном» цикле *for*. Ширина не больше длины, поэтому вложенный цикл конечный. Как только условие задачи выполнится, мы печатаем ответ на экране и завершаем процедуру Solve оператором *exit*.

Итак, длина составляла 6 ладоней, а ширина – 4:



Впрочем, нетрудно сообразить, что ¾ ширины равны 10 – 7 = 3 ладони, поэтому вся ширина равна 4 ладоням.

Индийские пчёлы

А эту задачу придумали в Древней Индии в то же время, что и задачу про длину и ширину в Древнем Вавилоне:

> *Пчёлы числом, равным квадратному корню из половины их числа во всём рое, сели на куст жасмина, 8/9 пчёл полетели назад к рою. И только одна пчела из того же роя кружилась над цветком лотоса, привлечённая жужжанием подруги, неосторожно угодившей в ловушку сладко благоухающего цветка.*
>
> *Сколько всего пчёл было в рое?*

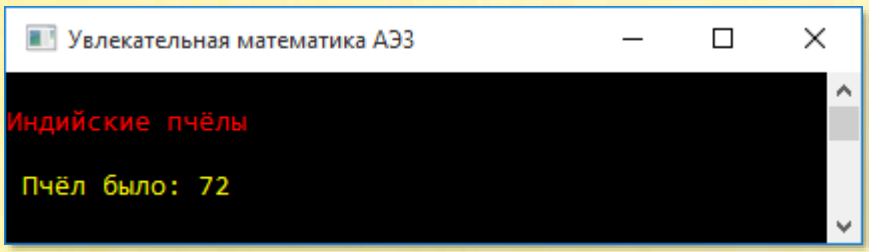
Так число пчёл – **целое**, то в рое должно быть столько пчёл, чтобы их число нацело делилось и на 2 и на 9, то есть на 18. Этого наблюдения нам вполне достаточно, чтобы решить задачу:

```
uses
   System;

// Увлекательная математика АЭЗ


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   for var b := 0 to integer.MaxValue do
```

```
    begin
      var bee := b * 18;
      var sqrtbee := Math.Sqrt(double(bee) / 2);
      if (sqrtbee + double(bee) * 8 / 9 + 2 = bee) then
      begin
        Console.WriteLine(' Пчёл было: ' + bee);
        Console.WriteLine();
        break;
      end
    end;
    Console.WriteLine();
end;


begin
    // заголовок окна:
    Console.Title := 'Увлекательная математика АЭЗ';
    Console.WriteLine('');
    Console.ForegroundColor := ConsoleColor.Red;
    Console.WriteLine('Индийские пчёлы');
    Console.ForegroundColor := ConsoleColor.Green;
    Console.WriteLine();

    Solve();

    Console.WriteLine();
    Console.ForegroundColor := ConsoleColor.Red;
end.
```

Для извлечения квадратного корня мы воспользовалтсь методом Sqrt из математического класса *Math*.

Пчёл в рое было 72:



Алгебраическое решение, приведённое в книге *Увлекательная математика*, значительно сложнее переборного!

Как указывает автор этой книги, 4000 лет назад математика была своеобразным видом спорта. Устраивались состязания по решению сложных задач, на которых присутствовали многочисленные зрители.

Пчелиная задача взята из учебного пособия по проведению математических конкурсов, причём в оригинале она изложена в *стихотворной* форме.

Китайские кролики и фазаны

Одна из самых известных исторических занимательных задач – про кроликов и фазанов – была известна ещё в Древнем Китае. В трактате *Киу-чанг – Девять отделов арифметики*, - написанном в 2637 году до нашей эры, имеется такая задача:

> *В клетке находится неизвестное число фазанов и кроликов. Известно только, что общее число голов – 35 и число ног – 94.*
>
> *Требуется узнать число фазанов и число кроликов.*

На сайте http://festival.1september.ru/articles/569698/ вы найдёте эту же задачу с подробным разбором её решения:

Сегодня на уроке мы вновь встретимся с вами с хорошо известной вам задачей про фазанов и кроликов (задача выводится на доску "В клетке находятся фазаны и кролики. Известно, что у них 35 голов и 94 ноги. Узнайте число фазанов и число кроликов"), но если раньше мы её решали арифметическим способом, то сегодня будем её решать с помощью уравнений и даже системы уравнений.

Поскольку ни число голов, ни число ног в программе не изменяется, то мы сохраним их в **константах**: HEADS - число *голов* и LEGS - число *ног*:

```
const
   // общее число голов:
   HEADS = 35;
   // общее число ног:
   LEGS = 94;
```

Максимальное число кроликов (*Rabbits*) находим делением общего числа ног на 4 (у кролика 4 ноги, или лапы):

```
   // макс. число кроликов:
   var maxRabbits := LEGS div 4;
```

Минимальное число кроликов равняется нулю – тогда все головы и ноги принадлежат только фазанам. Теперь мы должны перебрать все возможные числа для поголовья кроликов в цикле ***for***. Ясно, что переменная цикла *i* изменяется в диапазоне *0..maxRabbits*. Число фазанов (*Pheasants*) мы найдём, вычтя из общего числа голов число кроликов, то есть *i*.

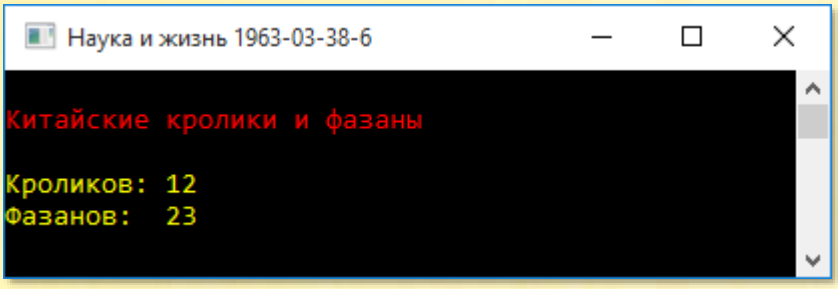
У *i* кроликов *i*4* ноги, а у *nPheasants* фазанов - *nPheasants*2* ног. Когда сумма ног совпадёт с требуемой (*LEGS*), мы печатаем решение.

В целом исходный код программы может быть таким:

```
uses
   System;

// Наука и жизнь 1963-03-38-6


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
const
   // общее число голов:
   HEADS = 35;
   // общее число ног:
   LEGS = 94;

begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   // макс. число кроликов:
   var maxRabbits := LEGS div 4;

   for var i := 0 to maxRabbits do
   begin
      // число фазанов:
      var nPheasants := HEADS - i;
      if (i * 4 + nPheasants * 2 = LEGS) then
         Console.WriteLine('Кроликов: ' + i + NewLine + 'Фазанов:  ' +
nPheasants);
   end;
   Console.WriteLine();
end;


begin
   // заголовок окна:
   Console.Title := 'Наука и жизнь 1963-03-38-6';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Китайские кролики и фазаны');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();
   //Solve2();
   Console.WriteLine();
```

```
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

Запускаем программу и читаем ответ на задачу: кроликов было 12, а фазанов – 23.



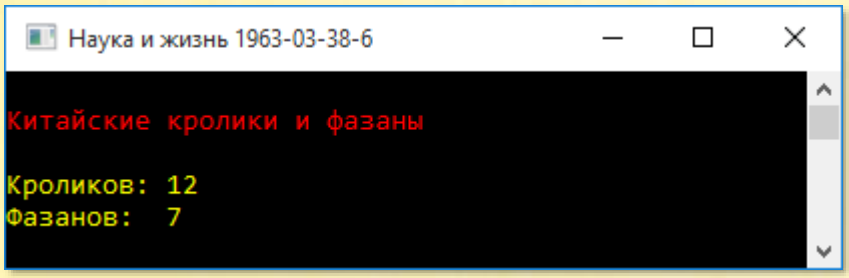
Эту древнюю задачу до сих пор решают в средней школе:

> *У фазанов и кроликов 62 ноги и 19 голов.*
>
> *Сколько фазанов и кроликов?*

Достаточно изменить значения констант – и ответ получен:

```
procedure Solve2();
const
   // общее число голов:
   HEADS = 19;
   // общее число ног:
   LEGS = 62;

begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   // макс. число кроликов:
   var maxRabbits := LEGS div 4;

   for var i := 0 to maxRabbits do
   begin
      // число фазанов:
      var nPheasants := HEADS - i;
      if (i * 4 + nPheasants * 2 = LEGS) then
```

```
            Console.WriteLine('Кроликов: ' + i + NewLine +
                              'Фазанов:  ' + nPheasants);
    end;
    Console.WriteLine();
end;
```

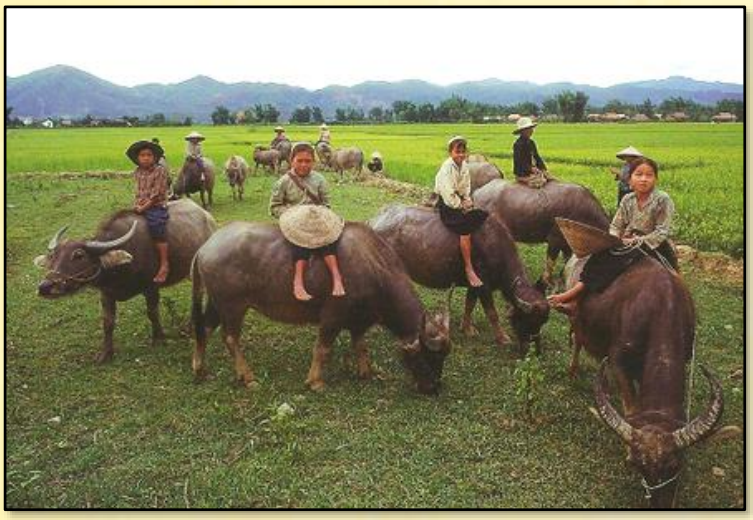


А вот как решали задачу про кроликов и фазанов сами китайцы:

Если бы в клетке были одни фазаны, то число ног было бы 70, а не 94. Следовательно, 24 лишних ноги принадлежат кроликам, по две на каждого. Кроликов – 12, фазанов – 23.

Иногда хочется быть древним китайцем!

Вьетнамские буйволы



Вот очень старая вьетнамская задача, которую старики-рисоводы любят задавать подрастающему поколению:

> *Для кормления 100 буйволов заготовили 100 охапок сена.*
>
> *Стоящий молодой буйвол съедает 5 охапок сена.*
> *Лежащий молодой буйвол съедает 3 охапки сена.*
> *Старые буйволы втроём съедают 1 охапку сена.*
>
> ***Сколько молодых буйволов стоят, сколько лежат и сколько буйволов старых?***

Решение задачи с буйволами не очень сильно отличается от решения кроличье-фазаньей.

Понятно, что буйволов каждого вида не меньше нуля и не больше – (100 поделить на число съедаемых охапок сена). Некоторую нервозность вносят старые буйволы, которых следует считать тройками.

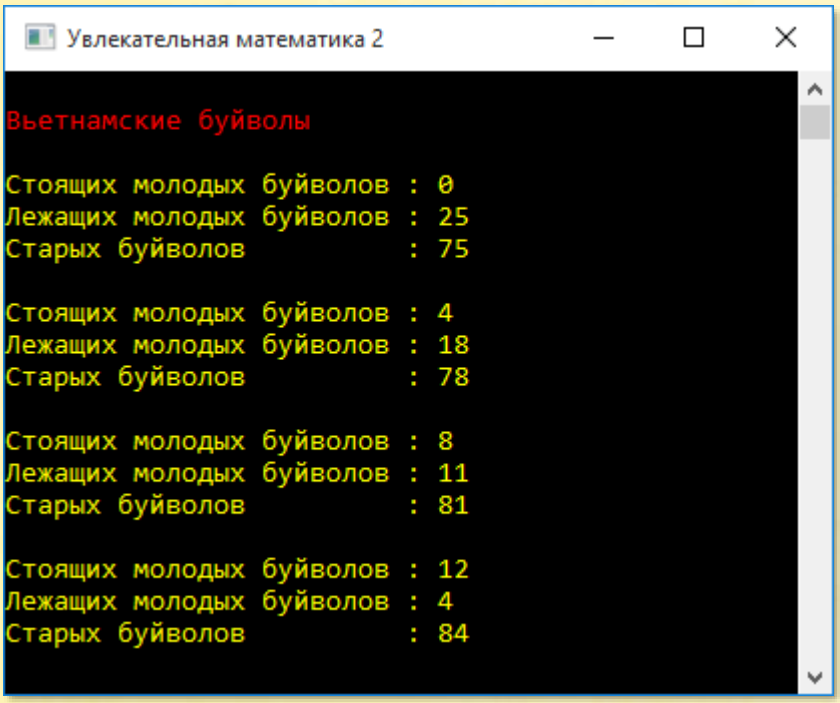
Когда общее число буйволов и съедаемых охапок сена будет по 100, мы печатаем ответ:

```
uses
```

```
   System;

// Увлекательная математика 2


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   for var buffaloS := 0 to 100 div 5 do
      for var buffaloL := 0 to 100 div 3 do
         for var bO := 0 to 100 div 3 do
         begin
            var buffaloO := bO * 3;
            if ((buffaloS + buffaloL + buffaloO = 100) and
                (buffaloS * 5 + buffaloL * 3 +
                 buffaloO div 3 = 100)) then
            begin
               Console.WriteLine('Стоящих молодых буйволов : '
                  + buffaloS);
               Console.WriteLine('Лежащих молодых буйволов : '
                  + buffaloL);
               Console.WriteLine('Старых буйволов          : '
                  + buffaloO);
               Console.WriteLine();
            end
         end;
   Console.WriteLine();

end;


begin
   // заголовок окна:
   Console.Title := 'Увлекательная математика 2';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Вьетнамские буйволы');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();
```

```
    Solve();

    Console.WriteLine();
    Console.ForegroundColor := ConsoleColor.Red;
end.
```

Задача имеет 4 решения, что вполне могло поставить вьетнамских недорослей в тупик.



Индийские обезьяны

Две очень старые индийские задачи про обезьян. Подобные задачи были известны в Индии ещё в XII веке.

Сколько обезьян в стаде, если квадрат одной пятой их без 3 скрылось в пещере, а одна залезла на дерево?

Эта задача легко решается с помощью квадратного уравнения. Но это в Индии – там тепло. А нам сподручнее решить задачу с помощью перебора:

```
uses
   System;

// Фольклор Д10


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   for var monkey := 0 to integer.MaxValue do
   begin
      var m5 := monkey div 5 - 3;
      if ((m5 > 0) and (m5 * m5 + 1 = monkey)) then
      begin
         Console.WriteLine(' Обезьян было: ' + monkey);
         break;
      end;
   end;
   Console.WriteLine();
end;


begin
```

```
    // заголовок окна:
    Console.Title := 'Математический фольклор. Задача Д10';
    Console.WriteLine('');
    Console.ForegroundColor := ConsoleColor.Red;
    Console.WriteLine('Индийские обезьяны');
    Console.ForegroundColor := ConsoleColor.Green;
    Console.WriteLine();

    Solve();

    Console.WriteLine();
    Console.ForegroundColor := ConsoleColor.Red;
end.
```

Условие

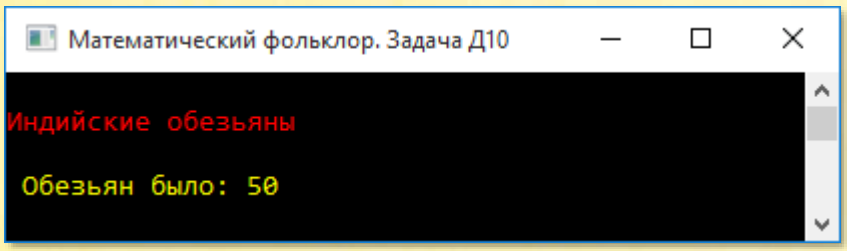
```
m5 * m5 + 1 = monkey
```

следует из текста задачи и представляет собой как раз квадратное уравнение. Оно имеет 2 корня – минус 5 и 50.

Условие

```
m5 > 0
```

игнорирует первый корень, поскольку число обезьян, скрывшихся в пещере, не может быть отрицательным. Остаётся второй корень – 50:



Индийские обезьяны 2

Квадрат восьмой части стада обезьян играют в роще, а остальные 12 – на горке.

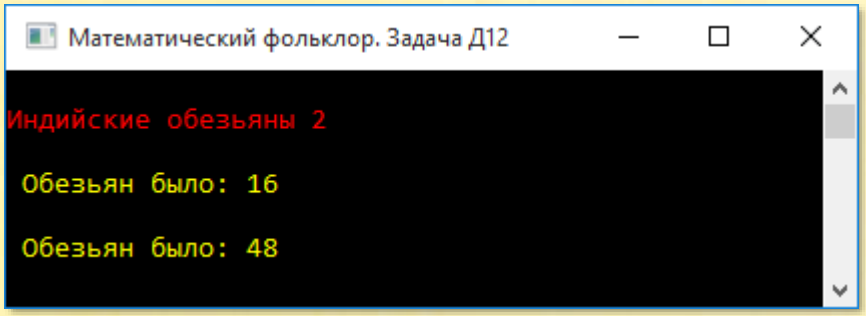
Сколько обезьян в стаде?

Поскольку задача опять решается с помощью квадратного уравнения, то нам нужно найти оба корня. Для этого в бесконечном цикле *while* мы считаем найденные решения:

```
uses
   System;

// Фольклор Д12


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
    Console.ForegroundColor := ConsoleColor.Yellow;

    // число решений:
    var variant := 0;
    var monkey := 0;
    while(true) do
    begin
        var m8 := monkey div 8;
```

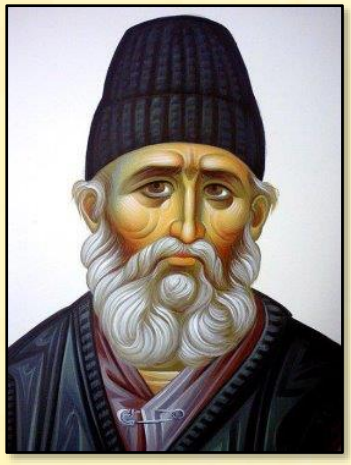
```
        if (monkey - m8 * m8 = 12) then
        begin
            Console.WriteLine(' Обезьян было: ' + monkey);
            Console.WriteLine();
            variant += 1;
            if (variant = 2) then
                break;
        end;
        monkey += 8;
    end;
    Console.WriteLine();
end;


begin
   // заголовок окна:
   Console.Title := 'Математический фольклор. Задача Д12';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Индийские обезьяны 2');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

На этот раз оба ответа правильные:



Жизнь Демохара

Множество занимательных задач связано с вычислением **возраста**. Одна из первых появилась в Греции в конце IV века:



Демохар четверть жизни прожил мальчиком, пятую часть – юношей, одну треть – зрелым мужчиной и 13 лет пожилым.

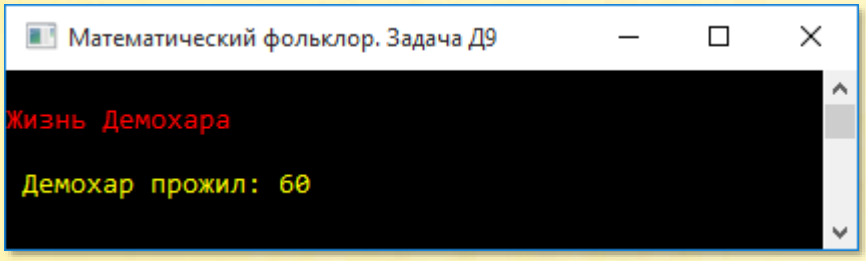
Сколько лет прожил Демохар?

Демохар прожил не меньше 13 лет (наверняка гораздо дольше, но перебор всё равно будет небольшим, поэтому мы можем удовлетвориться и этим явно заниженным возрастом). Так как в условии присутствует деление, то возраст лучше хранить в переменной типа *double*:

```
uses
   System;

// Математический фольклор. Задача Д9


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   for var l := 13 to integer.MaxValue do
   begin
      var let := double(l);
      if (let / 4 + let / 5 + let / 3 + 13 = let) then
      begin
         Console.WriteLine(' Демохар прожил: ' + let);
         Console.WriteLine();
         break;
      end
   end;
```

```
    Console.WriteLine();
end;


begin
    // заголовок окна:
    Console.Title := 'Математический фольклор. Задача Д9';
    Console.WriteLine('');
    Console.ForegroundColor := ConsoleColor.Red;
    Console.WriteLine('Жизнь Демохара');
    Console.ForegroundColor := ConsoleColor.Green;
    Console.WriteLine();

    Solve();

    Console.WriteLine();
    Console.ForegroundColor := ConsoleColor.Red;
end.
```

Демохар прожил 60 лет:



Если мы будем считать, что результат деления всегда **целое** число, то возраст Демохара должен нацело делиться на 4, 5 и 3. Минимальное число, удовлетворяющее этим требованиям, - 60. Следующее число – 120 – явно не по Демохару.

Греческие мешконосы

Эта задача приписывается Евклиду. Ей около 2300 лет!

Нагруженные осёл и мул идут очень медленно. Осёл жалуется на непосильную ношу, а мул отвечает:

- Что ты жалуешься? Если я возьму один твой мешок, то моя ноша станет в 2 раза тяжелее твоей. А если ты возьмёшь один мой мешок, то наши ноши будут равны.

По скольку мешков несли осёл и мул?

Эту задачу можно найти в книге *Увлекательная математика. Античные этюды. Задача 7*:

Однажды мула и осла нагрузили зерном. По дороге мул сказал ослу:

- Если бы ты уступил мне одну меру своего груза, то я нёс бы вдвое больше зерна, чем ты. А если бы я уступил тебе одну меру своего груза, то мы оба несли бы зерна поровну.

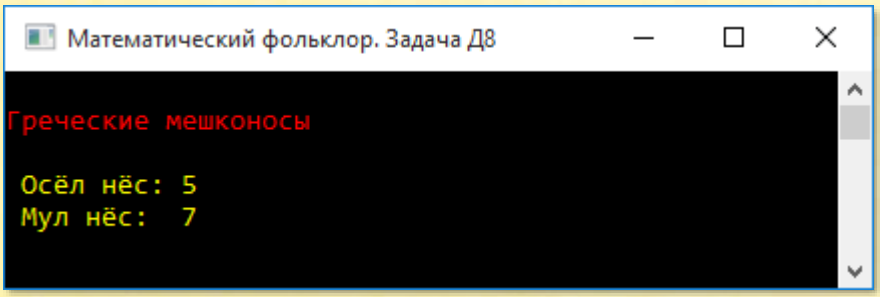
По скольку мер зерна нёс мул и сколько - осёл?

И осёл, и мул несли не менее 1 мешка (даже не меньше двух). Здравый смысл нам подсказывает, что мул нёс не более 100 мешков (наверняка гораздо меньше). Сколько мешков нёс осёл, нам знать необязательно, потому что мы будем добавлять ему мешки в «бесконечном» цикле ***for***:

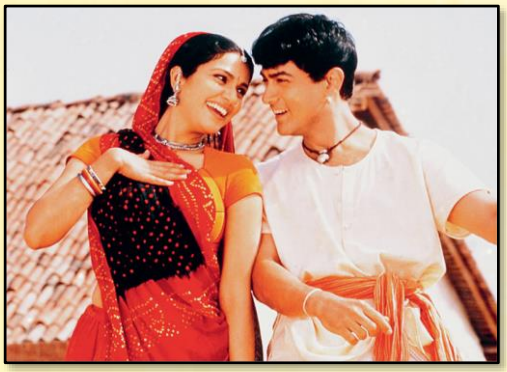
```
uses
   System;
```

```
// Фольклор Д8

// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   for var osel := 1 to integer.MaxValue do
      for var mul := 1 to 100 do
      begin
         var uslovie1 := mul + 1 = 2 * (osel - 1);
         var uslovie2 := mul - 1 = osel + 1;
         if (uslovie1 and uslovie2) then
         begin
            Console.WriteLine(' Осёл нёс: ' + osel);
            Console.WriteLine(' Мул нёс:  ' + mul);
            Console.WriteLine();
            exit;
         end
      end;
   Console.WriteLine();
end;

begin
   // заголовок окна:
   Console.Title := 'Математический фольклор. Задача Д8';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Греческие мешконосы');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

Ответ на задачу:



Индийское число



Условие задачи:

> *Если некоторое число умножить на 5, от произведения отнять его треть, остаток разделить на 10 и к полученному числу прибавить последовательно одну треть, одну вторую и одну четвёртую первоначального числа, то получится 68.*
>
> *Какое это число?*

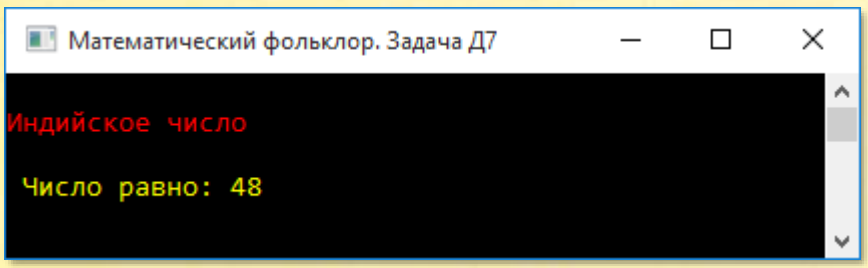
Мы можем только посочувствовать древним индийцам, у которых не было компьютеров, чтобы решать такие задачи.

А для компьютера эта задача очень простая – так же, как и для нас. Главное - правильно записать её условие:

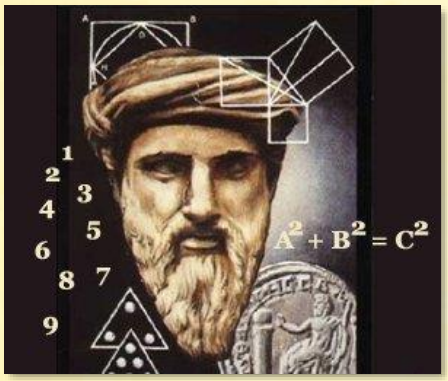
```
uses
   System;

// Фольклор Д7


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;
```

```
    for var num := 1 to integer.MaxValue do
      if ((num * 5 - num * 5 div 3) div 10 + num div 3 +
          num div 2 + num div 4 = 68) then
      begin
        Console.WriteLine(' Число равно: ' + num);
        Console.WriteLine();
        break;
      end;
    Console.WriteLine();
end;


begin
    // заголовок окна:
    Console.Title := 'Математический фольклор. Задача Д7';
    Console.WriteLine('');
    Console.ForegroundColor := ConsoleColor.Red;
    Console.WriteLine('Индийское число');
    Console.ForegroundColor := ConsoleColor.Green;
    Console.WriteLine();

    Solve();

    Console.WriteLine();
    Console.ForegroundColor := ConsoleColor.Red;
end.
```

Мы могли бы предположить, что искомое число должно делиться нацело на 2, 3 и 4 и тем значительно сократить перебор, но в данном случае этого не требуется:



Пифагорейское число



Знаменитого Пифагора спросили:

- Сколько учеников посещают твою школу и слушают твои беседы?

Пифагор ответил:

- Половина моих учеников изучает математику, четверть – музыку, седьмая часть проводит время в молчаливом размышлении, остальную часть составляют 3 ученика.

Сколько учеников было у Пифагора?

Эта задача напечатана и в книге *Увлекательная математика. Античные этюды. Задача 5. Пифагор Самосский (ок. 508-501 гг. до н.э.)*:

Поликрат (известный из баллады Шиллера тиран с острова Самос) однажды спросил на пиру у Пифагора, сколько у того учеников.

- Охотно скажу тебе, о Поликрат, - отвечал Пифагор. - Половина моих учеников изучает прекрасную математику, четверть исследует тайны вечной природы, седьмая часть молча упражняет силу духа, храня в сердце учение. Добавь ещё к ним трёх юношей, из которых Теон превосходит прочих своими способностями.

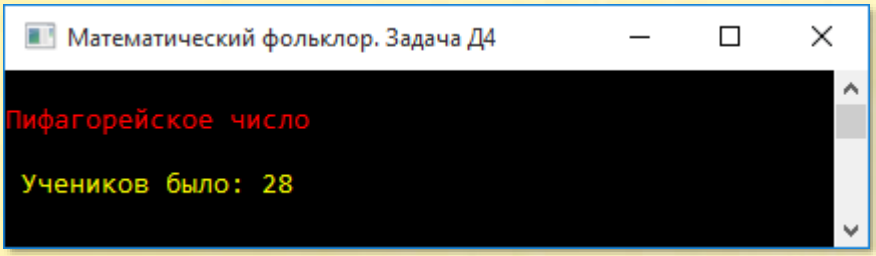
Сколько учеников было у Пифагора?

Так как число учеников нужно делить на 2, 4 и 7, то для переменной «бесконечного» цикла *for* следует выбрать тип *double*. Условие прекращения цикла легко выводится из текста задачи:

```
uses
   System;

// Фольклор Д4


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   for var u := 1 to integer.MaxValue do
   begin
      var uchenik := double(u);
      if (uchenik / 2 + uchenik / 4 + uchenik / 7 + 3 = uchenik)
then
      begin
         Console.WriteLine(' Учеников было: ' + uchenik);
         Console.WriteLine();
         break;
      end
   end;
   Console.WriteLine();
end;


begin
   // заголовок окна:
   Console.Title := 'Математический фольклор. Задача Д4';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Пифагорейское число');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();
```

```
    Console.WriteLine();
    Console.ForegroundColor := ConsoleColor.Red;
end.
```

У Пифагора было 28 учеников:



Задача легко решается в уме, если учесть, что число учеников должно быть кратно 4 и 7.

Индийский храм

Из четырёх посетителей храма второй дал в 2 раза больше монет, чем первый, третий – в 3 раза больше монет, чем второй, а четвёртый – в 4 раза больше монет, чем третий. Всего было дано 132 монеты.

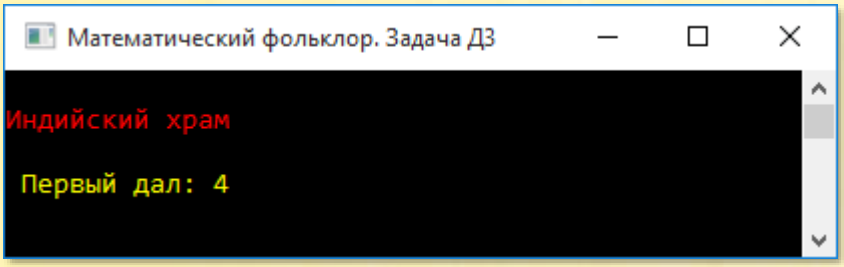
Сколько монет дал первый?

Мы будем увеличивать в «бесконечном» цикле *for* число монет первого посетителя храма. Число монет остальных посетителей легко найти с помощью умножения. Когда общая сумма достигнет 132, цикл завершается:

```
uses
   System;

// Фольклор ДЗ


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   for var monet := 1 to integer.MaxValue do
   begin
      if (monet + monet * 2 + monet * 2 * 3 +
          monet * 2 * 3 * 4 = 132) then
      begin
         Console.WriteLine(' Первый дал: ' + monet);
         Console.WriteLine();
         break;
      end
   end;
   Console.WriteLine();
end;


begin
   // заголовок окна:
   Console.Title := 'Математический фольклор. Задача ДЗ';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Индийский храм');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();
```

```
    Console.WriteLine();
    Console.ForegroundColor := ConsoleColor.Red;
end.
```

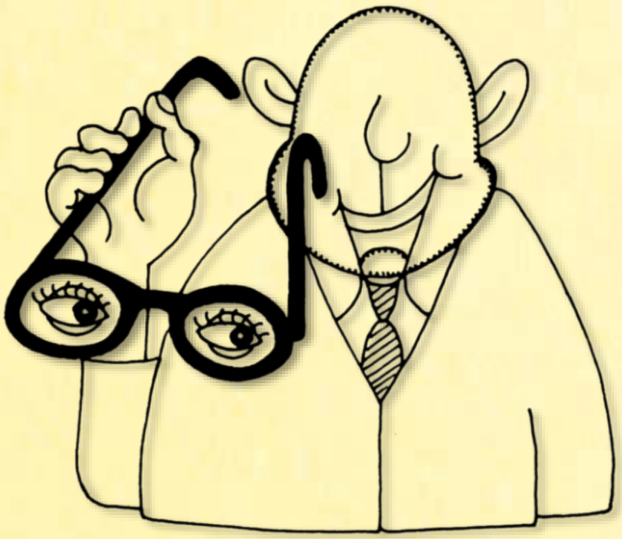
Первый посетитель дал 4 монеты:



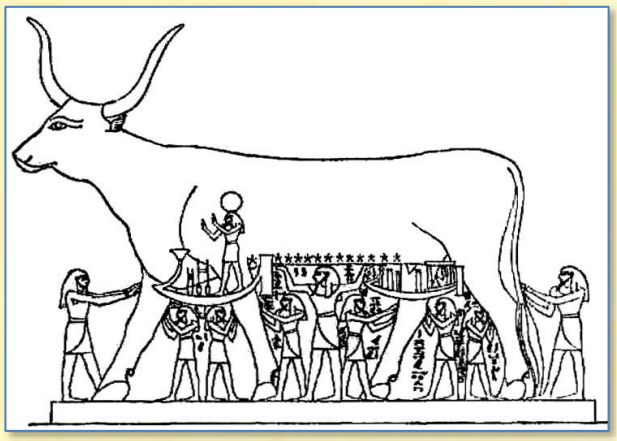
В Древней Греции такие задачи решали методом приведения к единице. Он напоминает наш перебор, но умнее.

Действительно, если первый посетитель даст одну монету, то общая сумма составит 33 монеты. Это в 4 раза меньше требуемой. Следовательно, во столько же раз больше первый посетитель должен был дать монет. То есть 4.

Как хорошо, что у древних греков не было компьютеров!



Египетские коровы



В этом проекте мы решим одну задачу из папируса Ринда (Райнда). Возраст папируса оценивается в 3700 лет. Он был найден англичанином Риндом в конце XIX века.

Но папирус – только часть более древнего математического труда третьего тысячелетия до нашей эры:

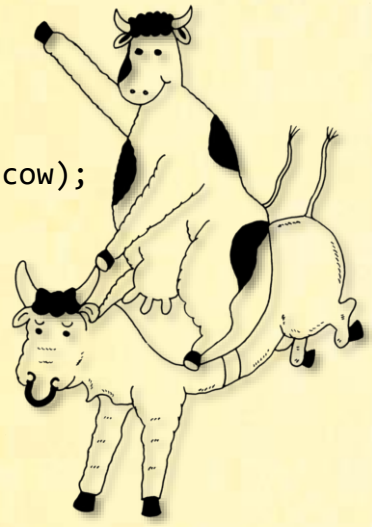
> *Некий математик насчитал на выгоне 70 коров.*
>
> *- Какую долю от всего стада составляют эти коровы? – спросил математик у пастуха.*
>
> *- Я выгнал пастись две трети от трети всего стада, - отвечал пастух.*
>
> ***Сколько голов скота насчитывается во всём стаде?***

Так как коровы паслись не консервированные, а цельные, то число голов в стаде должно делиться на 9.

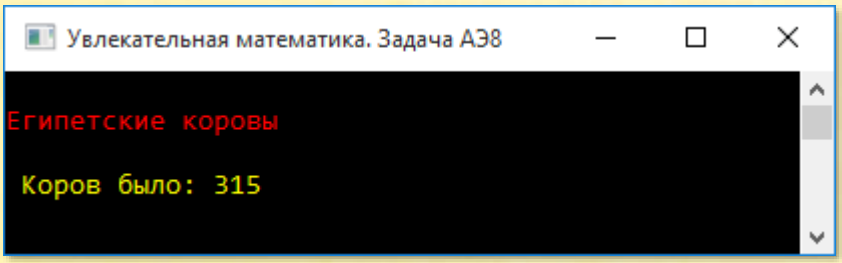
Задача легко решается и без перебора, поэтому мы применяем компьютер только из почтения к древности этой математической задачи:

```
uses
   System;
```

```
// Увлекательная математика АЭ8


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   var cow := 0.0;
   while(true) do
   begin
      if (cow * 2 / 3 * 1 / 3 = 70) then
      begin
         Console.WriteLine(' Коров было: ' + cow);
         Console.WriteLine();
         break;
      end;
      cow += 9;
   end;
   Console.WriteLine();
end;


begin
   // заголовок окна:
   Console.Title := 'Увлекательная математика. Задача АЭ8';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Египетские коровы');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```



А коров было много:



Римские адвокаты

Крючкотворная римская задача первого века до нашей эры:

Адвокаты в Древнем Риме имели обыкновение задавать друг другу задачи. Одна из таких задач гласит:

Некая вдова должна разделить оставшееся после смерти мужа наследство в размере 3500 динариев с ещё не родившимся ребёнком. По римским законам, если родится сын, то мать получает половину причитающейся ему доли, а в случае рождения дочери мать получает вдвое больше неё. У вдовы родились близнецы – сын и дочь.

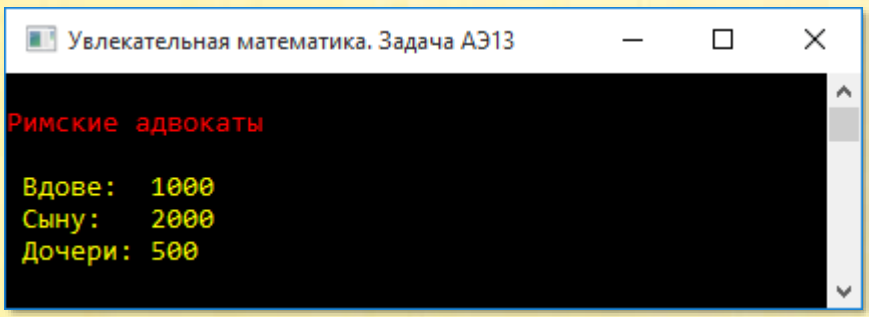
Как разделить наследство, чтобы все требования закона были соблюдены?

Мы полагаем, что задача решается в **целых** числах. Тогда легко найти долю вдовы в цикле ***for***. Совершенно очевидно, что она не может получить больше 3500 динариев (и даже значительно меньше). Тогда сын получает вдвое больше, а дочь вдове меньше:

```
uses
   System;

// Увлекательная математика АЭ13


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;
```

```
   for var vdova := 0 to 3500 do
   begin
      var syn := 2 * vdova;
      var doch := vdova div 2;
      if (vdova + syn + doch = 3500) then
      begin
         Console.WriteLine(' Вдове:  ' + vdova);
         Console.WriteLine(' Сыну:   ' + syn);
         Console.WriteLine(' Дочери: ' + doch);
         Console.WriteLine();
         break;
      end
   end;
   Console.WriteLine();
end;


begin
   // заголовок окна:
   Console.Title := 'Увлекательная математика. Задача АЭ13';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Римские адвокаты');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

Всё – мы разделили наследство по-честному, то есть по древнеримским законам:



Диофантово число

Задача древнегреческого математика Диофанта Александрийского, жившего в третьем веке:

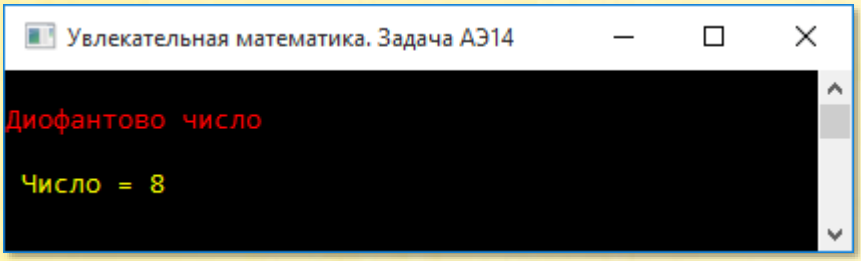
> *По двум данным числам 200 и 5 найти третье число, которое, если его умножить на одно из них, даёт полный квадрат, а если его умножить на другое число, даёт квадратный корень из этого квадрата.*

Мы можем найти заданное число перебором в «бесконечном» цикле ***for***, начиная с единицы. Полный квадрат найдём умножением 200 (но не 5!) на текущее значение переменной цикла. Для определения принадлежности произведения к полным квадратам мы напишем функцию IsQuadrat:

```
uses
   System;

// Увлекательная математика АЭ14

// ОПРЕДЕЛЯЕМ ЯВЛЯЕТСЯ ЛИ ЗАДАННОЕ ЧИСЛО
// ПОЛНЫМ КВАДРАТОМ
function IsQuadrat(num: int64): boolean;
begin
   var r := Trunc(Math.Sqrt(num));
```

```
   Result := r * r = num;
end;


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   for var num := 1 to integer.MaxValue do
   begin
      if (IsQuadrat(num * 200) and (Math.Sqrt(num * 200) = 5 *
num)) then
      begin
         Console.WriteLine(' Число = ' + num);
         Console.WriteLine();
         break;
      end
   end;
   Console.WriteLine();
end;


begin
   // заголовок окна:
   Console.Title := 'Увлекательная математика. Задача АЭ14';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Диофантово число');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

Искомое число равно 8:



Годится также число 0, но оно явно не древнегреческое.

Если не копировать полностью требование задачи, то условие можно записать проще:

```
if (5*num = Math.Sqrt(num*200))
```

Арабские голуби

Задача из арабских сказок *1001 ночь* (ночь 458), которые были написаны много столетий назад:

> *Стая голубей подлетела к высокому дереву. Часть голубей села на ветвях, а другая расположилась под деревом. Сидевшие на ветвях голуби говорят расположившимся внизу:*

- Если бы один из вас взлетел к нам, то вас стало бы втрое меньше, чем нас всех вместе, а если бы один из нас слетел к вам, то нас с вами стало бы поровну.

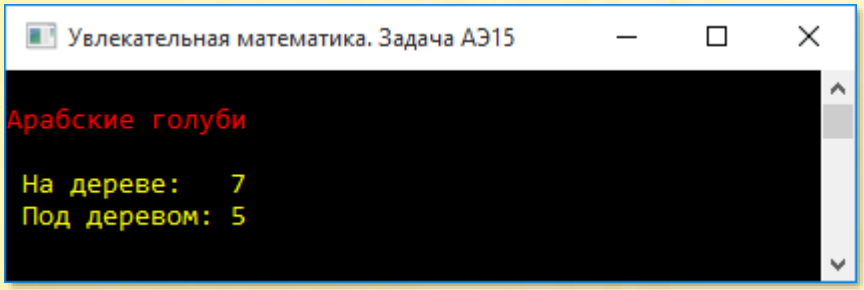
Сколько голубей сидело на ветвях и сколько под деревом?

Поскольку никаких намёков на число голубей мы не имеем, то только первый цикл *for* может быть «бесконечным». Во вложенном цикле мы полагаем, что голубей было меньше тысячи:

```
uses
   System;

// Увлекательная математика АЭ15


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   for var n := 0 to integer.MaxValue do
      for var p := 0 to 1000 - 1 do
      begin
         var na := double(n);
         var pod := double(p);
         if ((pod - 1 = (na + pod) / 3) and
               (na - 1 = pod + 1)) then
         begin
            Console.WriteLine(' На дереве:   ' + na);
            Console.WriteLine(' Под деревом: ' + pod);
            Console.WriteLine();
            exit;
         end
      end;
   Console.WriteLine();
end;


begin
   // заголовок окна:
   Console.Title := 'Увлекательная математика. Задача АЭ15';
```

```
    Console.WriteLine('');
    Console.ForegroundColor := ConsoleColor.Red;
    Console.WriteLine('Арабские голуби');
    Console.ForegroundColor := ConsoleColor.Green;
    Console.WriteLine();

    Solve();

    Console.WriteLine();
    Console.ForegroundColor := ConsoleColor.Red;
end.
```

На самом-то деле голубей было совсем немного:



Персидские яблоки

Задача из старинной персидской легенды *История Морадбальса*, включённой в сборник сказок *1001 ночь*. В ней мудрец задаёт молодой девушке такую задачу:

> *Одна женщина отправилась в сад собрать яблоки. Чтобы выйти из сада, ей нужно было пройти через 4 двери, у каждой из которых стоял стражник. Стражнику у первых дверей женщина отдала половину сорванных ею яблок. Дойдя до второго стражника, женщина отдала ему половину оставшихся яблок. Так же она поступила и с третьим стражником; а когда она поделилась яблоками со стражником у четвёртых дверей, то у неё осталось 10 яблок.*
>
> *Сколько яблок она собрала в саду?*

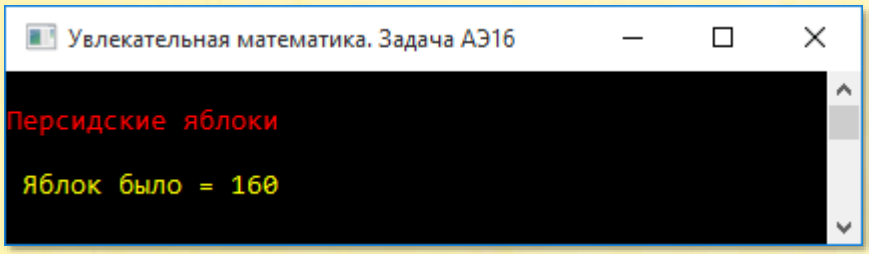
Поучительная задача, актуальная и в наши дни: чтобы пройти через двери, нужно делать взносы (или вклады).

Мы перебираем число яблок в «бесконечном» цикле ***for***, делимся ими со стражниками до тех пор, пока в результате не останется ровно десяток яблок:

```
uses
   System;

// Увлекательная математика АЭ16


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
```

```
begin
    Console.ForegroundColor := ConsoleColor.Yellow;

    for var apple := 1 to integer.MaxValue do
    begin
        var ost := apple;
        for var n := 1 to 4 do
        begin
            ost := ost div 2;
        end;
        if (ost = 10) then
        begin
            Console.WriteLine(' Яблок было = ' + apple);
            Console.WriteLine();
            exit;
        end
    end;
end;


begin
   // заголовок окна:
   Console.Title := 'Увлекательная математика. Задача АЭ16';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Персидские яблоки');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

А девушка, видимо, не в первый раз промышляла яблоками, так как нарвала их с большим запасом:



Кахунский папирус

Очень старая задача из *Кахунского папируса*:

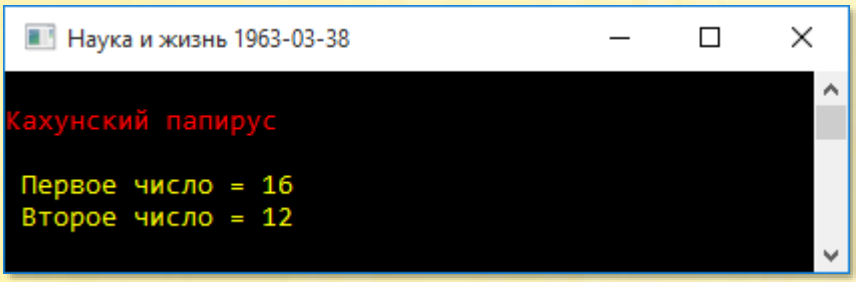
> *Отношение чисел равно 2 : 1½ , сумма квадратов – 400.*
>
> *Найти эти числа.*

Решение задачи очень простое. Главное – не запутаться в дробях:

```
uses
   System;

// Наука и жизнь 1963-03-38


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   for var num1 := 1 to integer.MaxValue do
   begin
      var num2 := num1 / 2.0 * 3 / 2;
      if (num1 * num1 + num2 * num2 = 400) then
      begin
         Console.WriteLine(' Первое число = ' + num1);
```

```
            Console.WriteLine(' Второе число = ' + num2);
            Console.WriteLine();
            break;
        end
    end;
    Console.WriteLine();
end;


begin
    // заголовок окна:
    Console.Title := 'Наука и жизнь 1963-03-38';
    Console.WriteLine('');
    Console.ForegroundColor := ConsoleColor.Red;
    Console.WriteLine('Кахунский папирус');
    Console.ForegroundColor := ConsoleColor.Green;
    Console.WriteLine();

    Solve();

    Console.WriteLine();
    Console.ForegroundColor := ConsoleColor.Red;
end.
```

Задача показывает, что древние математики не хуже нашего умели вычислять дроби и квадраты чисел:



Берлинский папирус

Ещё одна древневавилонская задача. На этот раз из *Берлинского папируса*:

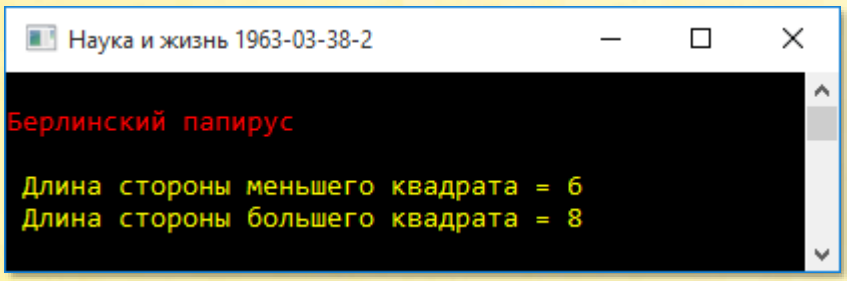
> *Если тебе будет сказано: разделить 100 квадратных локтей на 2 неизвестные части и ¾ стороны одной взять за сторону другой, дай мне каждую из неизвестных частей (иначе говоря, нужно разбить площадь в 100 квадратных локтей на 2 квадрата, стороны которых относились бы как 1 : 3/4.*

Так как стороны квадратов выражаются **целыми** числами, то длина стороны меньшего квадрата должна быть кратна трём:

```
uses
   System;

// Наука и жизнь 1963-03-38-2


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   var len := 0.0;
   while(true) do
   begin
      // длина второй стороны:
      var len2 := len * 4 / 3;
```

```
      if (len * len + len2 * len2 = 100) then
      begin
         Console.WriteLine(' Длина стороны меньшего квадрата = '
                           + len);
         Console.WriteLine(' Длина стороны большего квадрата = '
                           + len2);
         Console.WriteLine();
         break;
      end;
      len += 3;
   end;
   Console.WriteLine();
end;


begin
   // заголовок окна:
   Console.Title := 'Наука и жизнь 1963-03-38-2';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Берлинский папирус');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

Ответ на задачу:



Индийские квадраты

Индийская задача VIII века из *«Бахшалийской» рукописной арифметики*:

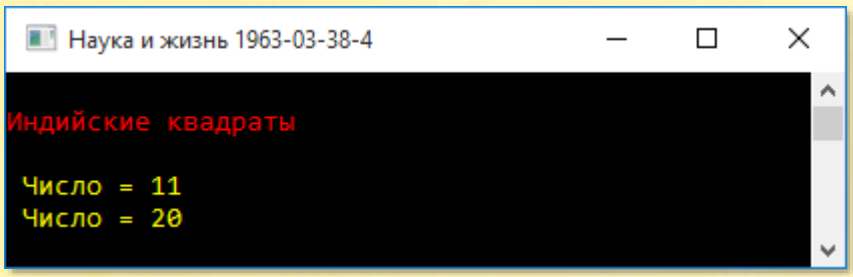
> *Найти число, которое от прибавления 5 или отнятия 11 обращается в полный квадрат.*

Искомое число не меньше 11, а верхнюю границу мы установим в сотню:

```
uses
   System;

// Наука и жизнь 1963-03-38-4

// ОПРЕДЕЛЯЕМ ЯВЛЯЕТСЯ ЛИ ЗАДАННОЕ ЧИСЛО
// ПОЛНЫМ КВАДРАТОМ
function IsQuadrat(num: int64): boolean;
begin
   var r := Trunc(Math.Sqrt(num));
   Result := r * r = num;
end;


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   for var num := 11 to 100 do
```

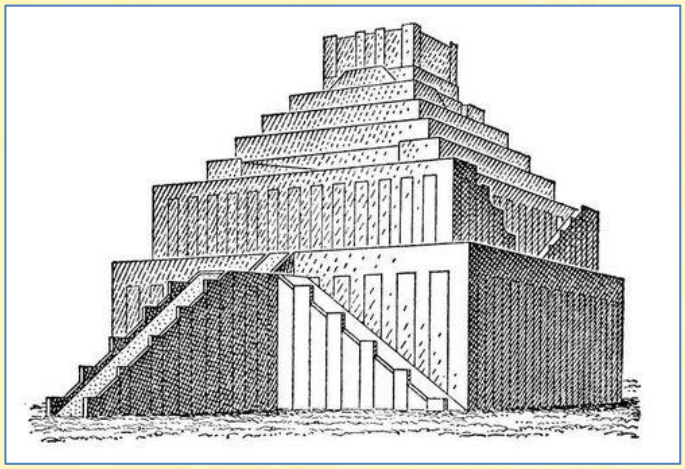
```
    begin
      var n5 := num + 5;
      var n11 := num - 11;
      if (IsQuadrat(n5) and IsQuadrat(n11)) then
      begin
        Console.WriteLine(' Число = ' + num);
      end;
    end;
    Console.WriteLine();
end;


begin
    // заголовок окна:
    Console.Title := 'Наука и жизнь 1963-03-38-4';
    Console.WriteLine('');
    Console.ForegroundColor := ConsoleColor.Red;
    Console.WriteLine('Индийские квадраты');
    Console.ForegroundColor := ConsoleColor.Green;
    Console.WriteLine();

    Solve();

    Console.WriteLine();
    Console.ForegroundColor := ConsoleColor.Red;
end.
```

Мы нашли 2 числа, удовлетворяющих условиям задачи:



При num = 11 мы получаем квадраты 16 и 0, а при num = 20 – 25 и 9.

Вавилонские квадраты и кубы



В Древнем Вавилоне пользовались 60-ричной системой счисления. Для записи чисел в такой системе требуется 60 цифр. Это много, но, с другой стороны, она очень удобна, так число 60 нацело делится на 2, 3, 4, 5, 6, 10, 12, 15, 20, 30 и 60. Недаром 60-ричная система счисления дожила до наших дней. Например, время и углы мы измеряем именно в этой системе счисления.

Математические вычисления в те далёкие времена были непростым делом, поэтому нередко встречаются задачи, в которых нужно просто вычислить значение какого-либо выражения. Дальше мы займёмся проверкой утверждений, записанных на двух глиняных табличках Сенкере:

1-я таблица.

На этой таблице помещены квадраты чисел от 1 до 60, выраженные по шестидесятеричной системе. Вот пример записи:

```
 1.21 есть квадрат  9
 2. 1 есть квадрат 11
 2.49 есть квадрат 13
 3.45 есть квадрат 15
 4.16 есть квадрат 16
25.21 есть квадрат 39
56. 4 есть квадрат 58
```

Проверьте справедливость этой записи. *(В записи чисел по шестидесятеричной системе числовые разряды отделены друг от друга точками.)*

2-я таблица.

Таблица содержит кубы чисел от 1 до 32:

```
 2. 5 есть куб  5
 3.36 есть куб  6
 5.43 есть куб  7
 8.32 есть куб 8
1.8.16 есть куб 16
9.6. 8 есть куб 32
```

Проверьте справедливость этой записи.

Для проверки первой и второй табличек мы напишем 2 функции - Solve2 и Solve3:

```
uses
   System;

// Наука и жизнь 1963-03-38-5

begin
   // заголовок окна:
   Console.Title := 'Наука и жизнь 1963-03-38-5';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Вавилонские квадраты и кубы');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve2();
   Solve3();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

Так как вавилонские числа записаны в 60-ричной системе, в которой разряды разделены точками, то для их хранения мы заведём **строковые массивы**. В пару к ним мы запишем и числа, которые нам предстоит возвести в квадрат или в куб:

```
// РЕШАЕМ ЗАДАЧУ
procedure Solve2();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   var vars60 := new string[]('1.21', '9',
                              '2.1', '11',
                              '2.49', '13',
                              '3.45', '15',
                              '4.16', '16',
                              '25.21', '39',
                              '56.4', '58');
```

Далее мы извлекаем пары строк из этих массивов и переводим в числа. Второе число – десятичное, поэтому мы конвертируем строку в число с помощью метода ToInt32 класса *Convert*. Для перевода строк с 60-ричными числами в десятичные такого метода у этого класса, естественно, нет, так что нам предстоит написать его самостоятельно:

```
   var i := 0;
   while(i < vars60.Count()) do
   begin
      // 60-ричное число:
      var s60 := vars60[i];
      var n60 := Get60(s60);
      // квадрат:
      var n := Convert.ToInt32(vars60[i + 1]);
      var n2 := n * n;
      if (n60 = n2) then
         Console.WriteLine(' 60-ричное число ' + s60 +
                           ' (' + n60 + ')' + ' равно ' + n +
                           ' в квадрате (' + n2 + ')')
      else
         Console.WriteLine(' 60-ричное число ' + s60 +
                           ' (' + n60 + ')' + ' не равно ' + n +
                           ' в квадрате (' + n2 + ')');
      i += 2;
```

```
   end;
   Console.WriteLine();
end;

procedure Solve3();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   var vars60 := new string[]('2.5', '5',
                              '3.36', '6',
                              '5.43', '7',
                              '8.32', '8',
                              '1.8.16', '16',
                              '9.6.8', '32');

   var i := 0;
   while(i < vars60.Count()) do
   begin
      // 60-ричное число:
      var s60 := vars60[i];
      var n60 := Get60(s60);

      // куб:
      var n := Convert.ToInt32(vars60[i + 1]);
      var n3 := n * n * n;
      if (n60 = n3) then
         Console.WriteLine(' 60-ричное число ' + s60 +
                           ' (' + n60 + ')' + ' равно ' + n +
                           ' в кубе (' + n3 + ')')
      else
         Console.WriteLine(' 60-ричное число ' + s60 +
                           ' (' + n60 + ')' + ' не равно ' + n +
                           ' в кубе (' + n3 + ')');

      i += 2;
   end;
   Console.WriteLine();
end;
```

Функция Get60 получает строку с 60-ричным числом и разбивает её на разряды:

```
function Get60(s60: string ): integer;
begin
```

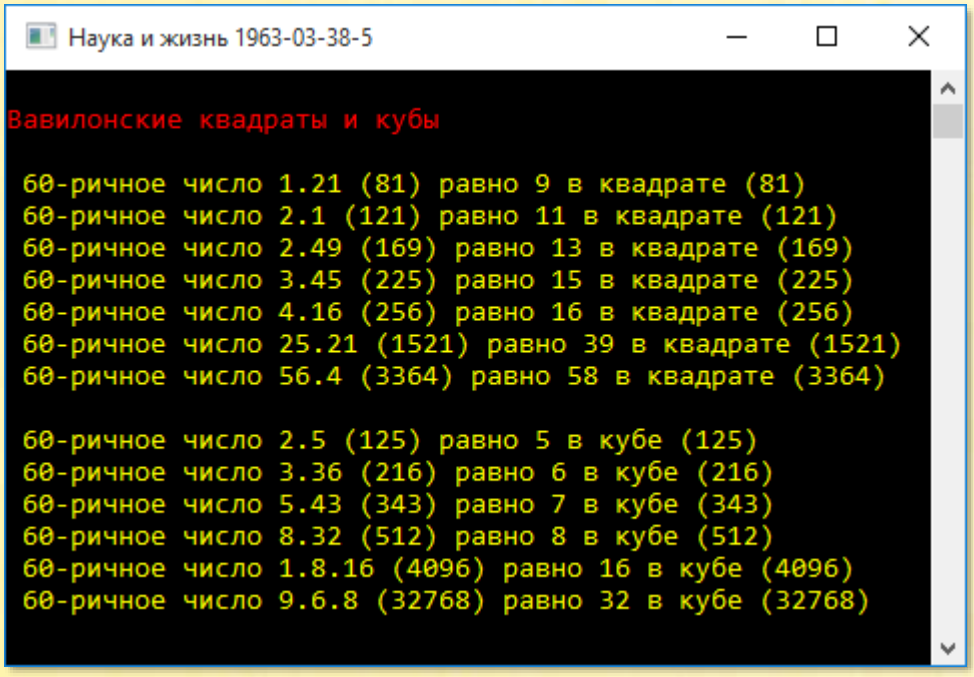
```
    Result := 0;
    var str := s60.Split(new char[]('.'));
```

В массиве строк **str** теперь хранятся подстроки с отдельными разрядами 60-ричного числа. Начиная с *последнего* разряда, мы умножаем десятичное число, соответствующее этому разряду, на 60 в степени номера разряда - 0, 1, 2, ..., то есть сначала на 1, затем на 60, затем на 3600 и так далее, пока разряды 60-ричного числа не закончатся.

Частичные произведения аккумулируются в переменной **Result**, значение которой и возвращается в вызывающую функцию:

```
  var k := 1;
   for var i := str.Count() - 1 downto 0 do
   begin
      Result += Convert.ToInt32(str[i]) * k;
      k *= 60;
   end;
end;
```

Что касается собственно проверки табличных данных, то рисунок ниже показывает и доказывает, что древние вавилоняне не зря портили глину – ни одной ошибки в их вычислениях мы не обнаружили!



СТАРЫЕ ЗАДАЧИ

Время неумолимо отсчитывает за веком век, занимательнее задачи становятся всё интереснее, изощрённее и многообразнее. Некоторые из них благополучно пережили века и до сих пор будоражат воображение и привлекают внимание любителей поломать голову.

В этой главе мы решим пару десятков любопытных задачек со всего света.

Чисто американская задача

Американцы любят деньги – доллары и центы. И почти все американские задачи только про деньги. Давайте попоможем им и решим старую (судя по ценам) задачу про деньги.

> *Мужчина потратил в магазине половину денег, которые имел при себе. Он заметил, что в кармане у него осталось столько центов, сколько долларов было до магазина, а долларов осталось половина от того, сколько центов он имел перед этим.*
>
> *Сколько денег было у мужчины сначала?*

Так как центов могло остаться от 0 до 99, то первоначально и долларов было от 0 до 99.

```
uses
   System;

// Фольклор Д45

begin
   // заголовок окна:
   Console.Title := 'Математический фольклор. Задача Д45';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
```

```
   Console.WriteLine('Чисто американская задача');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

Решаем задачу с помощью двух вложенных циклов *for*. Во внешнем цикле мы изменяем число долларов в заданном диапазоне, а во вложенном цикле - число центов:

```
// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   for var d := 0 to 99 do
      for var c := 0 to 99 do
      begin
```

По условию задачи, у американского мужчины осталось столько центов, сколько было долларов, а долларов – половина от прежних центов:

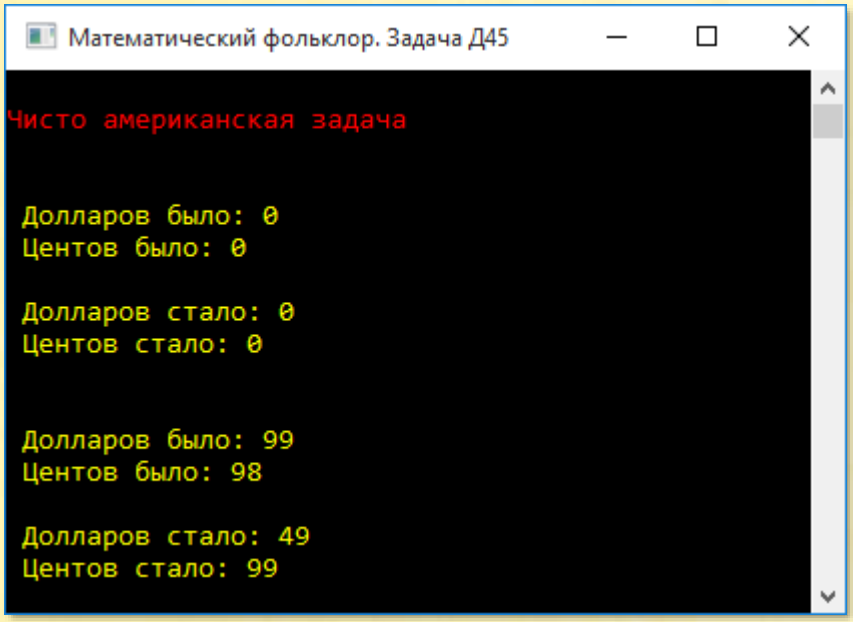
```
var c1 := d;
var d1 := c div 2;
```

После лёгкой прогулки по магазину у него осталась половина денег, что мы дальше и проверяем. Если книжное условие выполняется, мы печатаем ответ на задачу:

```
         if (100 * d + c = 2 * (100 * d1 + c1)) then
         begin
            Console.WriteLine(NewLine + ' Долларов было: ' + d +
```

```
                                NewLine + ' Центов было: ' + c);
            Console.WriteLine(NewLine + ' Долларов стало: ' +
                                d1 + NewLine +
                                ' Центов стало: ' + c1);
            Console.WriteLine();
          end
      end;
   Console.WriteLine();
end;
```

Довольно забавно, но задача имеет 2 решения:



Первый ответ более правдоподобен, но в книге приводится второй ответ.

Чисто французская задача

Французы мало уступают американцам по части любви к деньгам, но уже к своим. В те далёкие времена это были франки и сантимы.

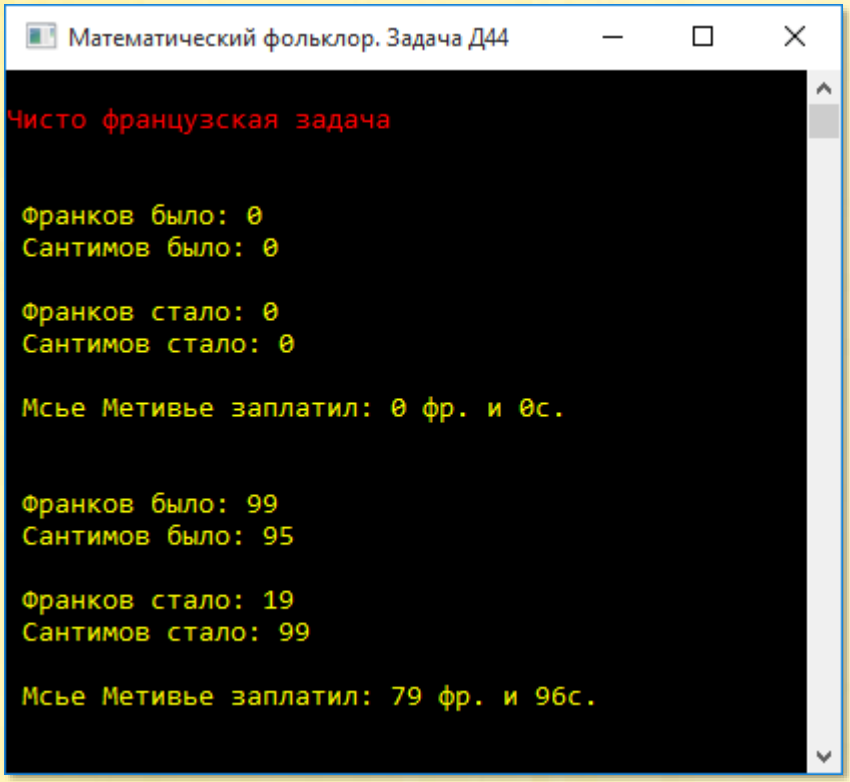
> *Мсье Метивье с женой решили отпраздновать годовщину своей свадьбы в ресторане.*
>
> *Уходя, мсье Метивье заплатил по счёту и заметил, что у него осталась одна пятая денег, которые были с собой. Причём сантимов осталось столько, сколько франков было вначале (1 франк = 100 сантимов), а оставшихся франков было в 5 раз меньше, чем сантимов вначале.*
>
> ***Какую сумму мсье Метивье заплатил в ресторане?***

Что касается арифметической подоплёки предложенной нам задачи, то достаточно выгодно обменять доллары и центы на франки и сантимы, а также учесть дефицит французского бюджета – и задача решена!

```
uses
   System;

// Фольклор Д44


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;
```

```
   for var f := 0 to 99 do
      for var c := 0 to 99 do
      begin
         var c1 := f;
         var f1 := c div 5;
         if (100 * f + c = 5 * (100 * f1 + c1)) then
         begin
            Console.WriteLine(NewLine + ' Франков было: ' + f +
                              NewLine + ' Сантимов было: ' + c);
            Console.WriteLine(NewLine + ' Франков стало: ' + f1 +
                              NewLine + ' Сантимов стало: ' + c1);
            var zaplatil := 4 * (100 * f1 + c1);
            Console.WriteLine(NewLine + ' Мсье Метивье заплатил: ' +
                              zaplatil div 100 +
                              ' фр. и ' + (zaplatil -
                              zaplatil div 100 * 100) + 'с.');
            Console.WriteLine();
         end
      end;
   Console.WriteLine();
end;


begin
   // заголовок окна:
   Console.Title := 'Математический фольклор. Задача Д44';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Чисто французская задача');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

Решение задачи слегка осложняется бухгалтерскими выкладками, но мы, умело сведя дебит с кредитом, тут же получаем расчёт счёта:



Поскольку мсье Метивье вряд ли осмелился бы пойти с женой в ресторан без франков и сантимов, то следует признать верным только **второй** ответ.

Репетитор

А вот русская задача вообще не про деньги, а, наоборот, про учёбу. И про то, как получить за это деньги.

В те далёкие времена, когда компьютеров ещё и в помине не было, школьники уже вовсю решали задачки по математике, что давалось им нелегко. Этот познавательный процесс хорошо описан Антоном Павловичем Чеховым в рассказе *Репетитор*.

Я надеюсь, что вы с удовольствием прочитаете этот рассказ Чехова (а за ним и многие другие) от начала до конца, поэтому перейдём непосредственно к **задаче**:

Теперь по арифметике... Берите доску. Какая следующая задача?
Петя плюёт на доску и стирает рукавом. Учитель берёт задачник и диктует:

- «Купец купил 138 арш. чёрного и синего сукна за 540 руб. Спрашивается, сколько аршин купил он того и другого, если синее стоило 5 руб. за аршин, а чёрное 3 руб.?»

Тяготы репетиторского труда мы пропускаем и читаем финал этой арифметической истории:

- И без алгебры решить можно, - говорит Удодов, протягивая руку к счётам и вздыхая. - Вот, извольте видеть...

Он щёлкает на счётах, и у него получается 75 и 63, что и нужно было.

Итак, ответ на задачу известен, и нам только и остаётся, что заменить старые счёты современными, то есть компьютером.

Обозначаем **длину** (*length*) сукна и **стоимость** (*cost*) соответствующими **константами**:

```
const
    // общая длина сукна в аршинах:
    LENGTH: integer = 138;
    // общая стоимость сукна в рублях:
    COST: integer  = 540;
```

На *COST* рублей можно купить не более **maxBlue** аршин синего сукна:

```
// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
    Console.ForegroundColor := ConsoleColor.Yellow;
```

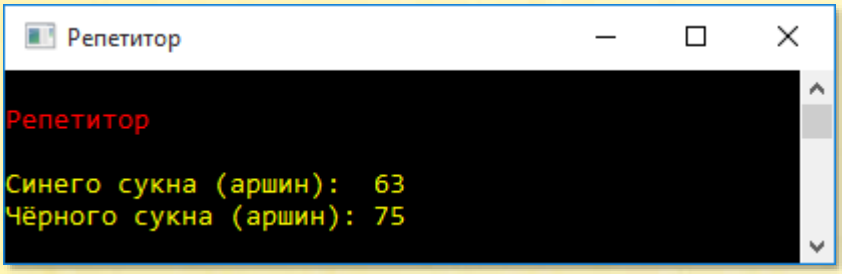
```
    // макс. длина синего сукна:
    var maxBlue := COST div 5;
```

Вы, конечно заметили, что эта задача принципиально ничем не отличается от старинной кроличье-фазаньей: заменив кроликов и фазанов синим и чёрным сукном, мы быстро выписываем весь переборный цикл *for*:

```
uses
   System;

// Репетитор

const
    // общая длина сукна в аршинах:
    LENGTH: integer = 138;
    // общая стоимость сукна в рублях:
    COST: integer  = 540;

// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
    Console.ForegroundColor := ConsoleColor.Yellow;

    // макс. длина синего сукна:
    var maxBlue := COST div 5;

    // решаем задачу:
    for var i := 0 to maxBlue do
    begin
        //длина чёрного сукна:
        var lenBlack  := LENGTH - i;
        if (i * 5 + lenBlack * 3 = COST) then
        begin
            Writeln('Синего сукна (аршин):  ' + i);
            Writeln('Чёрного сукна (аршин): ' + lenBlack);
        end;
    end;
    Console.WriteLine();
end;
```

```
begin
    // заголовок окна:
    Console.Title := 'Репетитор';
    Console.WriteLine('');
    Console.ForegroundColor := ConsoleColor.Red;
    Console.WriteLine('Репетитор');
    Console.ForegroundColor := ConsoleColor.Green;
    Console.WriteLine();

    Solve();

    Console.WriteLine();
    Console.ForegroundColor := ConsoleColor.Red;
end.
```

Запускаем программу и получаем верный ответ:



Как видите, решение таких задач на компьютере – дело техники, а вот на счётах – это уже искусство...

Американские цыпочки

В очередной американской задаче мы снова встречаемся с долларами. На этот раз с одинокими – без центов. Несколько скрашивают излишне меркантильную картину цыпочки, уточки и гусачки.

На вопрос, сколько стоит товар, продавец ответил:

- 3 цыплёнка и 1 утка стоят столько, сколько 2 гуся.
1 цыплёнок, 2 утки и 3 гуся вместе стоят 25 долларов. Причём каждый цыплёнок, утка и гусь стоят целое число долларов.

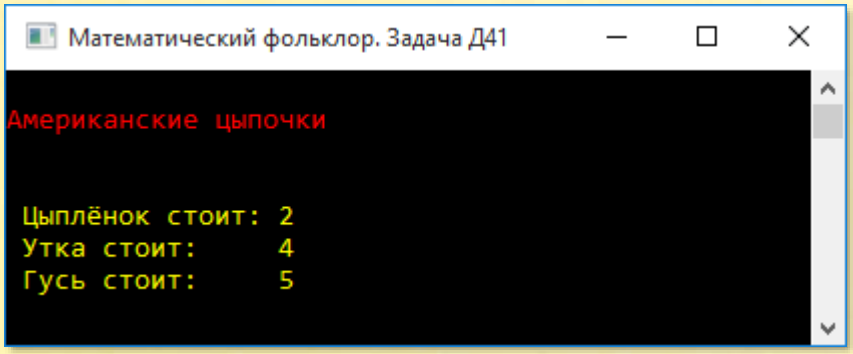
Сколько стоит каждая птица?

Из последнего равенства следует, что **цыплёнок** стоит не дороже 25 долларов, **утка** – не дороже 12 долларов и **гусь** – не дороже 8 долларов. Три вложенных цикла *for* – и мы узнаем цену американским цыпочкам:

```
uses
   System;

// Фольклор Д41


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
    Console.ForegroundColor := ConsoleColor.Yellow;

    for var cyplenok := 0 to 25 do
        for var gus := 0 to 8 do
            for var utka := 0 to 12 do
            begin
                if ((3 * cyplenok + utka = 2 * gus) and
```

```
                    (cyplenok + 2 * utka + 3 * gus = 25)) then
                begin
                    Console.WriteLine(NewLine+' Цыплёнок стоит: '
                                      + cyplenok +
                            NewLine+' Утка стоит:     ' + utka +
                            NewLine+' Гусь стоит:     ' + gus);
                    Console.WriteLine();
                end
            end;
    Console.WriteLine();
end;


begin
   // заголовок окна:
   Console.Title := 'Математический фольклор. Задача Д41';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Американские цыпочки');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

А вот и цены:



Американское наследство

Снова американская задача про американские деньги:

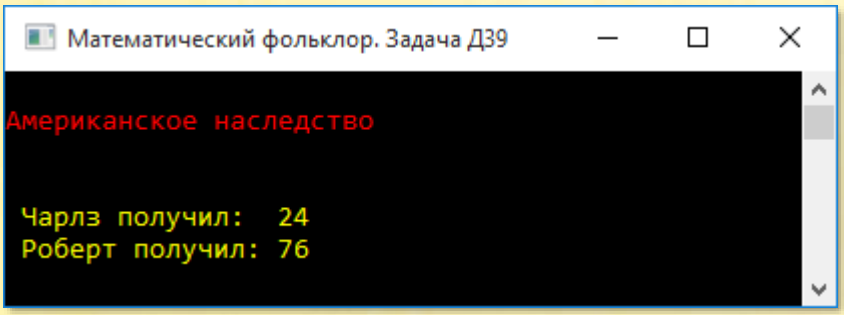
> *Отец оставил сыновьям Чарлзу и Роберту 100 долларов.*
>
> *Если одну треть части Чарлза вычесть из одной четверти части Роберта, то останется 11 долларов.*
>
> *Сколько долларов получил каждый из братьев?*

Обычно делёж наследства – процесс сложный и неприятный, но сотню долларов мы легко поделим – не поровну, но по-честному:

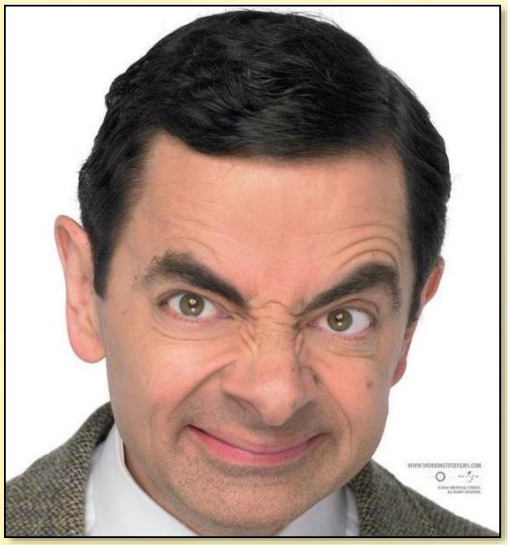
```
uses
   System;

// Фольклор Д39


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   for var Charles := 0 to 100 do
   begin
```

```
      var Robert := 100 - Charles;
      if (Robert div 4 - Charles div 3 = 11) then
      begin
         Console.WriteLine(NewLine + ' Чарлз получил:  ' +
                           Charles + NewLine +
                           ' Роберт получил: ' + Robert);
      end
   end;
   Console.WriteLine();
end;


begin
   // заголовок окна:
   Console.Title := 'Математический фольклор. Задача Д39';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Американское наследство');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

А вот и юридический отчёт о проделанной нами работе:



Английский юмор



Известно, что англичане хитрее американцев и всех остальных европейцев, поэтому и задачи у них с хитринками и подвохами.

В дом очаровательной мисс Чарити Локайер пришёл мистер Марк Девис, чтобы просить её руки. Его сопровождал товарищ. Мисс Локайер пригласила их в гостиную, подала 3 пустые чашки для чая и сахарницу, в которой было 10 кусочков сахара. Кавалеры помогали в сервировке стола. Мисс Локайер заявила, что отнесётся серьёзно к предложению мистера Девиса при одном условии: если он распределит 10 кусочков сахара по трём чашкам так, чтобы в каждой чашке было нечётное число кусков.

После некоторого смущения и краткого размышления мистер Девис нашёл верное решение. Какое?

Подобные задачи были известны в Англии ещё в XVII веке. Встречаются варианты со шляпами и корзинами.

Так как чашек было 3 – нечётное число, то чётное число кусочков сахара – 10 – нельзя между ними «распределить» без русской смекалки.

То есть после осахаривания чашек одну из них – с нечётным числом кусочков сахара – следует поставить в чашку с чётным числом кусочков сахара. Тогда в ней также будет нечётное число кусочков рафинада.

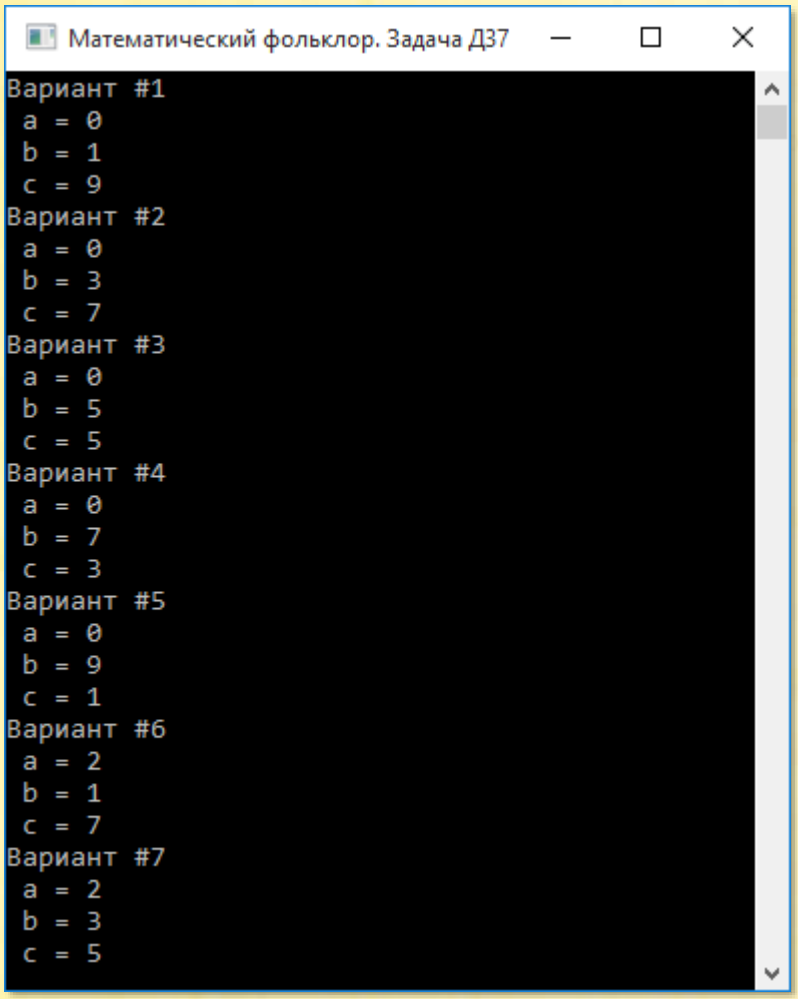
Обозначим чашку с чётным числом кусочков сахара буквой **a**, тогда остальные две чашки – с нечётным числом кусочков сахара – логично обозначить буквами **b** и **c**.

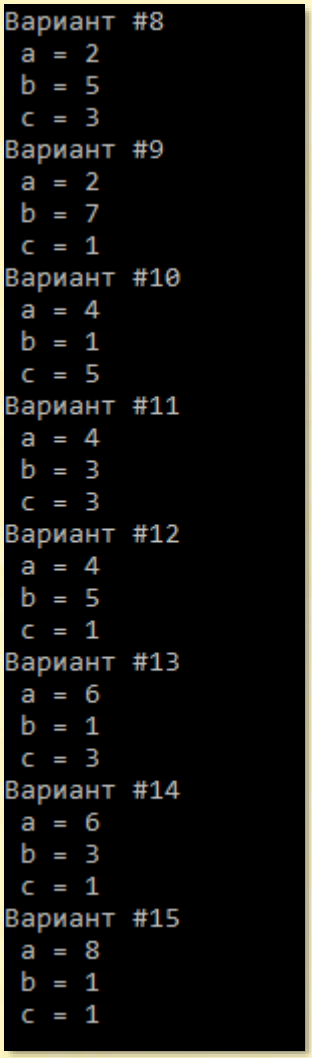
Из условия задачи следует, что число кусочков сахара в чашке **a** изменяется от 0 до 10, а в чашке **b** - от 1 до 10 – a. На чашку **c** приходятся оставшиеся кусочки сахара:

```
uses
   System;

// Фольклор Д37


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   var variant := 0;
   var a := 0;
   while(a <= 10) do
   begin
```

```
      var b := 1;
      while(b <= 10 - a) do
      begin
         var c := 10 - a - b;
         variant += 1;
         Console.WriteLine('Вариант #' + variant);
         Console.WriteLine(' a = ' + a +
                           NewLine + ' b = ' + b +
                           NewLine + ' c = ' + c);
         Console.WriteLine();
         b += 2;
      end;
      a += 2;
   end
end;


begin
   // заголовок окна:
   Console.Title := 'Математический фольклор. Задача Д37';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Английский юмор');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

Существует 15 способов решения сахарной задачи коварной, но очаровательной мисс Локайер:





Русские яблоки

А вот задачка про яблочки наливные!

Крестьянка несла на базар корзину яблок. Первому покупателю она продала половину всех яблок и ещё половину яблока, второму – половину от оставшихся яблок и ещё половину яблока, третьему – половину от нового остатка и ещё половину яблока и т.д. Когда шестой купил половину оставшихся яблок и ещё половину яблока, то яблок в корзине не осталось.

Сколько яблок было у крестьянки, если каждый купил целое число яблок?

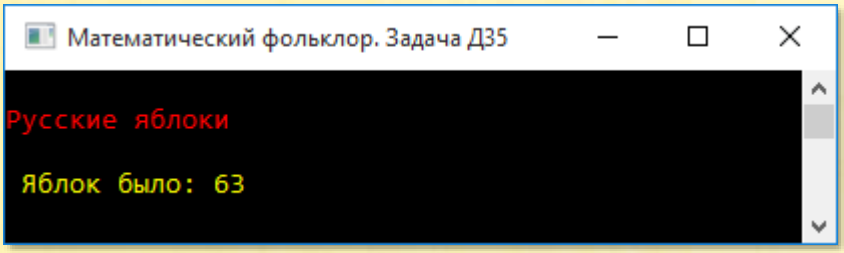
У болгар есть аналогичная задача, но про **груши**.

Условие задачи настолько **циклично**, что мы просто обязаны применить цикл *for*, в котором 6 покупателей поочерёдно приобретают яблоки. Число яблок нам неизвестно, поэтому мы перебираем их во внешнем «бесконечном» цикле *for*, начиная с единицы. Как только крестьянка продаст всю корзину яблок, мы печатаем ответ и заканчиваем решение задачи:

```
uses
   System;

// Математический фольклор. Задача Д1


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
    Console.ForegroundColor := ConsoleColor.Yellow;

    for var apple := 1 to integer.MaxValue do
    begin
        var ost := double(apple);
        for var i := 1 to 6 do
        begin
            ost := (ost - 1)/2;
        end;
        if (ost = 0) then
        begin
            Console.WriteLine(' Яблок было: ' + apple);
            Console.WriteLine();
```

```
            exit;
        end
    end
end;


begin
   // заголовок окна:
   Console.Title := 'Математический фольклор. Задача Д35';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Русские яблоки');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

А ответ такой:



С задачей легко справиться вручную, если решать её **ретроспективно**, то есть начиная с конца.

Действительно, шестому покупателю досталась половина яблока и ещё столько же, то есть ровно 1 яблоко. После этого корзина опустела. Пятый купил в 2 раза больше + 1 яблоко, то есть 3. Четвёртый – 3*2 + 1 = 7. И так далее – до победного начала.

Американские яблоки

На этот раз американцы делят не наследство в долларах, а фрукты в яблоках:

> *Четыре семейства – Смит, Браун, Джонсон и Робинсон - имеют вместе 8 детей. В каждом семействе по одному мальчику и одной девочке.*
>
> *Однажды детям раздали 32 яблока. Ани получила 1 яблоко, Бетти – 2, Кэт – 3 и Мэри – 4 яблока.*
>
> *Оказалось, что Гарри Смит и его сестра взяли яблок поровну, Том Браун получил яблок в 2 раза больше, чем его сестра, Билл Джонсон получил в 3 раза больше, чем его сестра, а Джек Робинсон получил яблок в 4 раза больше своей сестры.*
>
> ***Определить фамилии девочек.***

Так как мы пока не знаем фамилий девочек, то диапазон числа яблок сестры каждого из братьев – **1..4**.

Тогда **Гарри** получил 1 * (1..4) яблока. А **вместе с сестрой** – 1 *(1..4 + 1).

Аналогично для остальных братьев и сестёр:

Том получил 2 * (1..4) яблока. А **вместе с сестрой** – 2 *(1..4 + 1).
Билл получил 3 * (1..4) яблока. А **вместе с сестрой** – 3 *(1..4 + 1).
Джек получил 4 * (1..4) яблока. А **вместе с сестрой** – 4 *(1..4 + 1).

Все вместе они получили:

1 *(1..4 + 1) + 2 *(1..4 + 1) + 3 *(1..4 + 1) + 4 *(1..4 + 1) = 32 яблока (1)

Таким образом, нам нужно выписать 4 вложенных цикла ***for*** и вычислять сумму полученных яблок до тех пор, пока она не станет равна 32.

Мы также должны учесть, что все девочки получили **разное** число яблок.

```
// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   for var SisterApple1 := 1 to 4 do
      for var SisterApple2 := 1 to 4 do
      begin
         if (SisterApple2 = SisterApple1) then
            continue;
         for var SisterApple3 := 1 to 4 do
         begin
            if ((SisterApple3 = SisterApple2) or
                (SisterApple3 = SisterApple1)) then
               continue;
            for var SisterApple4 := 1 to 4 do
            begin
               if ((SisterApple4 = SisterApple3) or
                   (SisterApple4 = SisterApple2) or
                   (SisterApple4 = SisterApple1)) then
                  continue;
               var apples := 1 * (SisterApple1 + 1) +
                             2 * (SisterApple2 + 1) +
                             3 * (SisterApple3 + 1) +
                             4 * (SisterApple4 + 1);
               if (apples = 32) then
                  Console.WriteLine('Задача решена!');
            end
         end
      end;
   Console.WriteLine();
end;
```

Если вы запустите программу, то получите сообщение, что задача решена. Но теперь нам нужно определить ещё фамилии девочек.

Запишем имена девочек и их фамилии в строковые массивы:

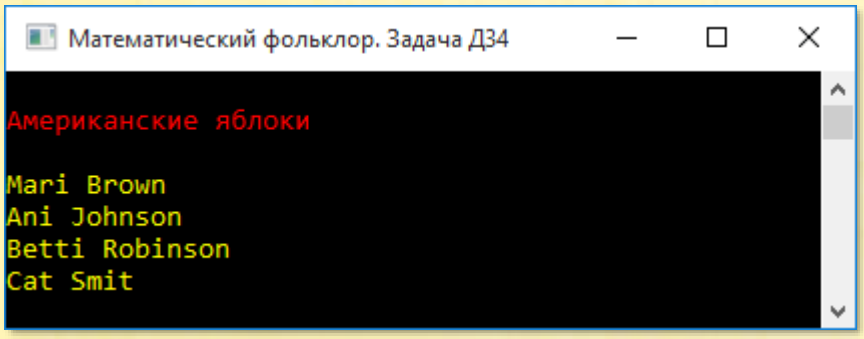
```
// имена девочек:
var SisterName := new string[]('', 'Ани', 'Бетти', 'Кэт', 'Мэри' );
// фамилии:                          Гарри    Том       Билл       Джек
var FamilyName := new string[]('', 'Смит', 'Браун', 'Джонсон', 'Робинсон' );
```

Фамилии в массиве FamilyName должны соответствовать кратности полученных братьями яблок. Именно так мы использовали имена братьев при выводе формулы (1).

Теперь мы легко напечатаем ответ на задачу в понятном виде:

```
if (apples = 32) then
begin
   //Console.WriteLine('Задача решена!');
   Console.WriteLine(SisterName[SisterApple1] + ' ' + FamilyName[1]);
   Console.WriteLine(SisterName[SisterApple2] + ' ' + FamilyName[2]);
   Console.WriteLine(SisterName[SisterApple3] + ' ' + FamilyName[3]);
   Console.WriteLine(SisterName[SisterApple4] + ' ' + FamilyName[4]);
end
```

Действительно, сестра Гарри получила *SisterApple1* яблок. Значит, её имя - *SisterName[SisterApple1]*. У неё такая же фамилия, как и у брата, то есть *FamilyName[1]*. Точно так же мы находим фамилии остальных девочек:



Задачу можно решить и несколько иначе:

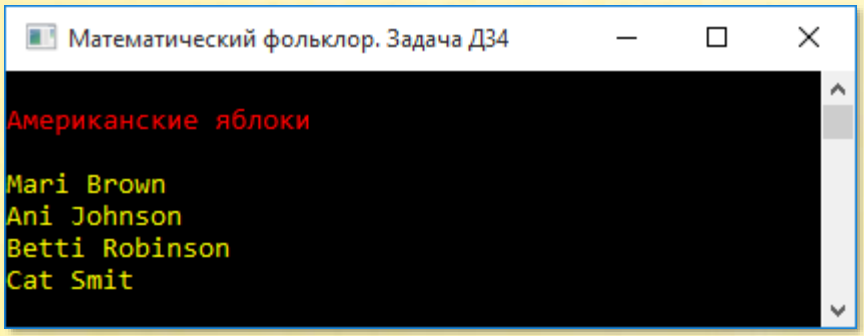
```
uses
   System;

// Фольклор Д34

begin
   // заголовок окна:
   Console.Title := 'Математический фольклор. Задача Д34';
   Console.WriteLine('');
```

```
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Американские яблоки');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   //Solve();
   Solve2();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.


// имена девочек:
type
   Names = (Ani, Betti, Cat, Mari );
// РЕШАЕМ ЗАДАЧУ
procedure Solve2();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;
   // имена девочек:
   var SisterName := new Dictionary<Names, string>();
   SisterName.Add(Names.Ani, 'Ani');
   SisterName.Add(Names.Betti, 'Betti');
   SisterName.Add(Names.Cat, 'Cat');
   SisterName.Add(Names.Mari, 'Mari');

   // число яблок у девочек:
   var apple := new Dictionary<string, integer>();
   apple.Add('Ani', 1);
   apple.Add('Betti', 2);
   apple.Add('Cat', 3);
   apple.Add('Mari', 4);

   // кратность яблок у мальчиков:
   var krat := new Dictionary<string, integer>();
   krat.Add('Гарри', 1);
   krat.Add('Том', 2);
   krat.Add('Билл', 3);
   krat.Add('Джек', 4);

   for var Brown := Names.Ani to Names.Mari do
      for var Johnson := Names.Ani to Names.Mari do
      begin
         if (Johnson = Brown) then
```

```
      continue;
   for var Robinson := Names.Ani to Names.Mari do
   begin
      if ((Robinson = Johnson) or (Robinson = Brown)) then
         continue;
      for var Smit := Names.Ani to Names.Mari do
      begin
         if ((Smit = Robinson) or (Smit = Johnson) or
                  (Smit = Brown)) then
            continue;
         // Brown -->
         // сестру звали:
         var name := SisterName[Brown];
         // у неё было яблок:
         var appleS := apple[name];
         // брата звали Том
         // он получил яблок:
         var appleB := appleS * krat['Том'];
         // вместе они получили:
         var appleSB1 := appleS + appleB;

         // Johnson -->
         // сестру звали:
         name := SisterName[Johnson];
         // у неё было яблок:
         appleS := apple[name];
         // брата звали Билл
         // он получил яблок:
         appleB := appleS * krat['Билл'];
         // вместе они получили:
         var appleSB2 := appleS + appleB;

         // Robinson -->
         // сестру звали:
         name := SisterName[Robinson];
         // у неё было яблок:
         appleS := apple[name];
         // брата звали Джек
         // он получил яблок:
         appleB := appleS * krat['Джек'];
         //вместе они получили:
         var appleSB3 := appleS + appleB;

         // Robinson -->
         // сестру звали:
```

```
                name := SisterName[Smit];
                // у неё было яблок:
                appleS := apple[name];
                // брата звали Джек
                // он получил яблок:
                appleB := appleS * krat['Гарри'];
                // вместе они получили:
                var appleSB4 := appleS + appleB;

                if (appleSB1 + appleSB2 + appleSB3 +
                          appleSB4 = 32) then
                begin
                   Console.WriteLine(SisterName[Brown] +
                                                ' Brown');
                   Console.WriteLine(SisterName[Johnson] +
                                                ' Johnson');
                   Console.WriteLine(SisterName[Robinson] +
                                                ' Robinson');
                   Console.WriteLine(SisterName[Smit] +
                                                ' Smit');
                   Console.WriteLine();
                end
             end
          end
       end
end;
```

Результат, естественно, вы получите тот же самый, что и раньше, но на «латинском» языке:



Американцы известны своим пристрастием к пересъёмкам французских фильмов на свой лад и со своими актёрами. Не избежала этой печальной участи и **французская задача** (*Увлекательная математика*, 1):

Во время летнего пикника четыре супружеские пары выпили 32 бутылки лимонада. Жёны выпили: Жанна – 1 бутылку, Жаклин – 2 бутылки, Колетта – 3 бутылки и Анетта – 4 бутылки. Мужья не уступили жёнам: месье Пон выпил столько же, сколько его жена, Месье Дюбуа – вдвое больше своей жены, месье Пейзан – втрое и месье Фонтен – вчетверо больше своих жён.

Как зовут мадам Пон, Дюбуа, Пейзан и Фонтен.

Свинская задача

Трое молодых датчан – Нильс, Клаас и Корнелиус отправились на базар со своими жёнами. Жён звали Геертринг, Катрин и Анна. Все шесть купили свиней, а когда вернулись домой, то оказалось, что все купили столько свиней, сколько крон стоила одна свинья.

При этом Нильс купил на 23 свиньи больше, чем Катрин, а Клаас купил на 11 свиней больше, чем Геертринг. Кроме того, каждый муж потратил на 63 кроны больше своей жены.

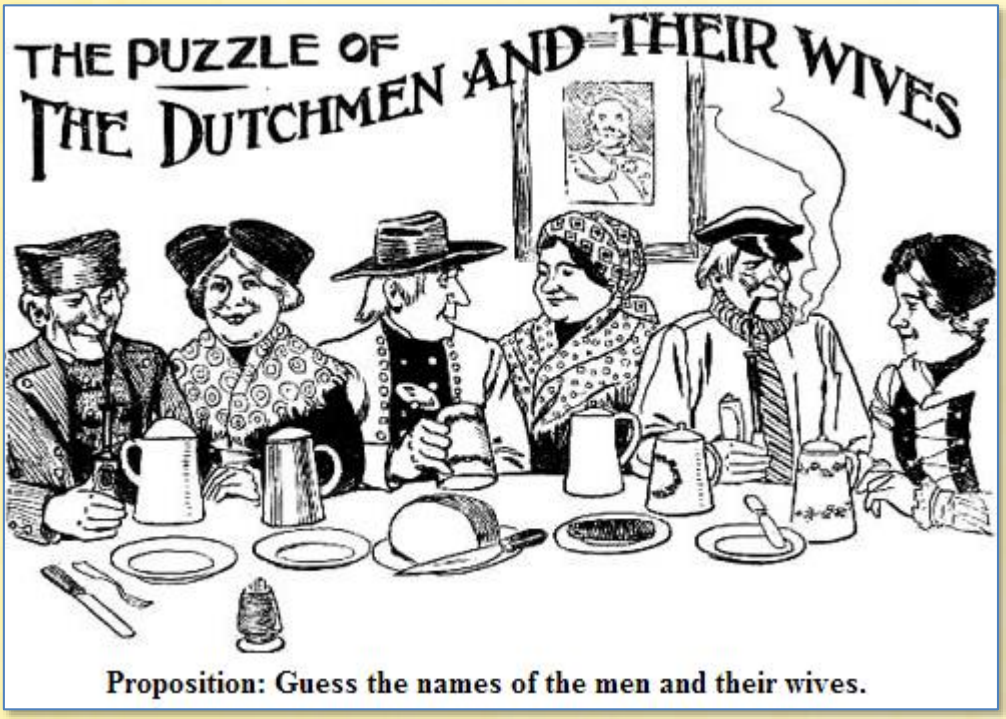
Как звали жену каждого из трёх датчан?

Вас не должны смущать древние имена некоторых персонажей этой задачи. Она появилась в Дании ещё в XVIII веке и остаётся популярной на протяжении нескольких веков. Например, эта задача напечатана в журнале *Popular Science:*

Answer to last month's P-O-T-M. Three Dutchmen and their wives went hog-shopping Each bought as many hogs as the price he paid per hog. Claas bought 11 more than Geertring, and Hendrik 23 more than Catrun. Cornelius and Anna are the other two. Husbands spent $63 more than wives. Match the couples.

Clearly this is a problem involving squares of whole numbers (no half-hogs allowed!). And clearly Hendrik is not married to Geertring nor Claas to Catrun (11 and 23 would have to divide evenly into 63 in that case). But you can write an equation for each married couple of the form $a^2 - 63 = b^2$. For if a is the price Hendrik paid per hog, we know he bought exactly a hogs and his total outlay was $\$a^2$. Similarly, if his wife paid $\$b$ per hog she spent a total of $\$b^2$. Now the simplest approach is to run through a table of squares and mark off pairs of numbers whose squares differ by 63. (You have an upper limit in this at 32—after that, consecutive squares differ by amounts larger than 63.) But right off the bat you find a qualifying pair: 1 and 8 ($64 - 1 = 63$). Furthermore, add 11 to 1 (using another clue) and you get 12, which is half of another qualifying pair: 12 and 9 ($144 - 81 = 63$). Now will the 9 work out with the third clue? Hendrik bought 23 more hogs than Catrun. $23 + 9 = 32$ which does qualify: $32^2 - 63 = 31^2$. Furthermore, no other pairs of numbers satisfy all conditions. So Claas is married to Catrun, Cornelius to Geertring, and Hendrik to Anna.

Её можно найти и в сборнике задач *Cyclopedia of Puzzles Сэма Лойда:*



Мы можем предположить, что ни одна жена не купила больше 100 свиней. Если эта гипотеза не приведёт нас к решению задачи, то мы расширим диапазон поиска.

Итак, согласно нашему предположению, Катрин купила от 1 до 100 свиней и заплатила за них costKatrin крон:

```
uses
   System;

// Фольклор ДЗ3

begin
   // заголовок окна:
   Console.Title := 'Математический фольклор. Задача ДЗ3';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Свинская задача');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();
```

```
   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   var Name := new List<string>();
   var Names := new List<string>();

   var Katrin := 0;
   for Katrin := 1 to 100 do
   begin
      // Katrin заплатила:
      var costKatrin := Katrin * Katrin;
```

Нам известно, что Нильс купил на 23 свиньи больше, чем Катрин, и заплатил за них costNils крон:

```
// Нильс купил:
var Nils := Katrin + 23;
// Нильс заплатил:
var costNils := Nils * Nils;
```

Геертринг также купила от одной до ста свиней, заплатив за них costGeertring крон:

```
var Geertring := 0;
 for Geertring := 1 to 100 do
 begin
    // Geertring заплатила:
    var costGeertring := Geertring * Geertring;
```

Клаас купил на 11 свиней больше, чем Геертринг, и заплатил costClaas крон:

```
// Клаас купил:
var Claas := Geertring + 11;
// Клаас заплатил:
var costClaas := Claas * Claas;
```

И наконец, Анна купила от 1 до 100 свиней и заплатила costAnna крон:

```
var Anna := 0;
 for Anna := 1 to 100 do
 begin
    // Anna заплатила:
    var costAnna := Anna * Anna;
```

Нильс заплатил на 63 кроны больше своей жены, и это должна быть одна из трёх женщин. Если при текущих значениях переменных *costNils*, *costKatrin*, *costGeertring* и *costAnna* это условие не выполняется, значит, нужно продолжать перебор:

```
  if ((costNils <> costKatrin + 63) and
      (costNils <> costGeertring + 63) and
      (costNils <> costAnna + 63)) then
              continue;
```

Аналогичное условие должно выполняться и для Клааса:

```
  if ((costClaas <> costKatrin + 63) and
      (costClaas <> costGeertring + 63) and
      (costClaas <> costAnna + 63)) then
              continue;
```

Если оба эти условия выполняются, то мы нашли для Нильса и Клааса их жён. Для дальнейших проверок и печати ответа нам понадобятся 2 списка:

```
var Name := new List<string>();
   var Names := new List<string>();
```

В **первый** мы поместим имена уже найденных жён, во **второй** – информацию для ответа.

Перед каждой проверкой мы очищаем оба списка:

```
Name.Clear();
Names.Clear();
```

Мы точно знаем, что при текущих значениях затрат на приобретение свиней Нильс заплатил на 63 кроны больше одной из женщин. Давайте установим её имя:

```
if (costNils = costKatrin + 63) then
begin
   Name.Add('Катрин');
   Names.Add('Катрин - жена Нильса');
end
else if (costNils = costGeertring + 63) then
begin
   Name.Add('Геертринг');
   Names.Add('Геертринг - жена Нильса');
end
else
begin
   Name.Add('Анна');
   Names.Add('Анна - жена Нильса');
end;
```

Теперь в список **Name** мы занесли имя предполагаемой жены Нильса.

Аналогичные разыскные мероприятия мы проводим и в отношении Клааса:

```
if ((costClaas = costKatrin + 63) and
   (not Name.Contains('Катрин'))) then
begin
   //Console.WriteLine(' Katrin - жена Клааса');
   Name.Add('Катрин');
   Names.Add('Катрин - жена Клааса');
end
```

```
  else if ((costClaas = costGeertring + 63) and
           (not Name.Contains('Геертринг'))) then
  begin
     //Console.WriteLine(' Geertring - жена Клааса');
     Name.Add('Геертринг');
     Names.Add('Геертринг - жена Клааса');
  end
  else if (not Name.Contains('Анна')) then
  begin
     //Console.WriteLine(' Анна - жена Клааса');
     Name.Add('Анна');
     Names.Add('Анна - жена Клааса');
  end;
```

После чего Нильс и Клаас должны получить имена своих жён. Поскольку жёны у них разные, то в списке имён *Name* должно оказаться ровно 2 записи:

```
if (Name.Count <> 2) then
    continue;
```

Печатаем на экране уже достоверно известную нам информацию:

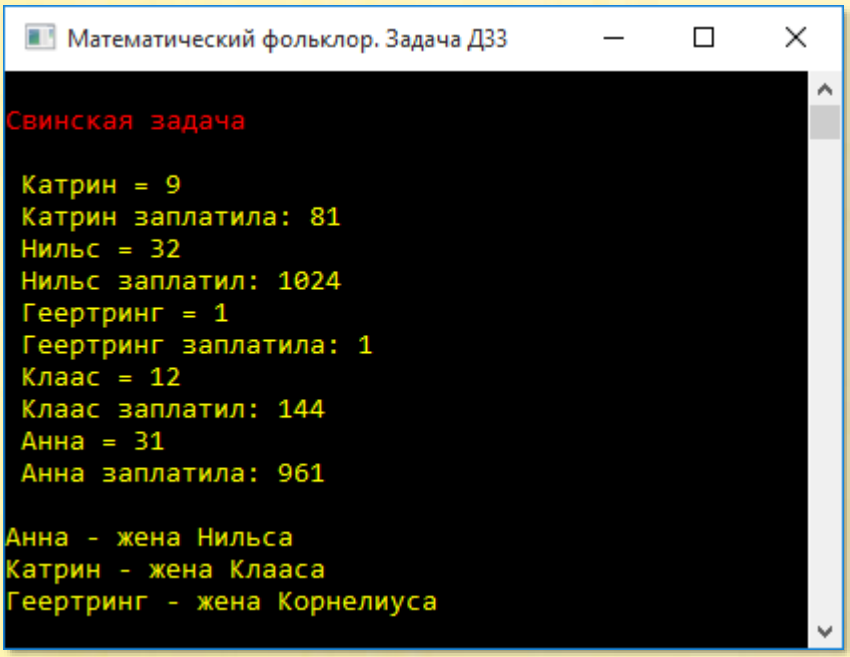
```
Console.WriteLine(' Катрин = ' + Katrin);
Console.WriteLine(' Катрин заплатила: ' + costKatrin);
Console.WriteLine(' Нильс = ' + Nils);
Console.WriteLine(' Нильс заплатил: ' + costNils);

Console.WriteLine(' Геертринг = ' + Geertring);
Console.WriteLine(' Геертринг заплатила: ' + costGeertring);
Console.WriteLine(' Клаас = ' + Claas);
Console.WriteLine(' Клаас заплатил: ' + costClaas);

Console.WriteLine(' Анна = ' + Anna);
Console.WriteLine(' Анна заплатила: ' + costAnna);
Console.WriteLine();
```

И последнее, что мы должны сделать, - найти жену для Корнелиуса. Это легко сделать, ведь её имя отсутствует в списке *Name*.

```
  if (not Name.Contains('Анна')) then
     Names.Add('Анна - жена Корнелиуса')
  else if (not Name.Contains('Геертринг')) then
     Names.Add('Геертринг - жена Корнелиуса')
  else
     Names.Add('Катрин - жена Корнелиуса');
```

Печатаем ответ на задачу:

```
            foreach var s in Names do
                Console.WriteLine(s);

             Console.WriteLine();
          end
       end
    end;
    Console.WriteLine();
end;
```



Вот такие задачи решали в XVIII веке!

Турецкие долгожители

Маленькое горное село славилось своими долгожителями. Особо уважали старого Исхана, имевшего детей, внуков, правнуков и праправнуков. Всего их вместе с Исханом было 2801. Праправнуки были ещё маленькие и не имели детей, а все остальные имели по одинаковому числу детей, и все дети были живы и здоровы.

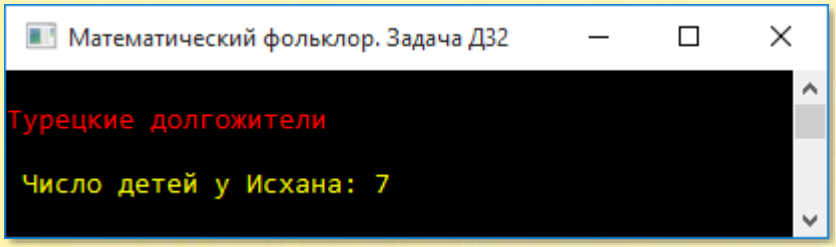
Сколько детей имел старый Исхан?

Задача чисто программистская и легко решается простым перебором в «бесконечном» цикле *for*:

```
uses
   System;

// Фольклор Д32


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
    Console.ForegroundColor := ConsoleColor.Yellow;

    for var n := 1 to integer.MaxValue do
    begin
        // n - число детей у Исхана
        // n * n - число внуков у Исхана
        // n * n * n - число правнуков у Исхана
        // n * n * n * n - число праправнуков у Исхана
        // всего - 2800 человек без Исхана:

        if (n + n * n + n * n * n + n * n * n * n = 2800) then
        begin
```

```
            Console.WriteLine(' Число детей у Исхана: '  + n);
            Console.WriteLine();
            break;
        end
    end;
    Console.WriteLine();
end;


begin
   //заголовок окна:
   Console.Title := 'Математический фольклор. Задача Д32';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Турецкие долгожители');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

Все детишки Исхана аккуратно пересчитаны:



Проспись и пей!

Наконец-то настоящая русская задача про вино!

> *Ивану, Петру и Василию необходимо разделить 7 бочек, полных вина, 7 бочек, наполненных до половины, и 7 пустых бочек.*
>
> ***Как им выполнить это, чтобы каждый получил по одинаковому количеству вина и равному числу бочек?***

> У болгар также имеется задача по эти винные бочки.

Из условия задачи явно следует, что каждый из русских иванов получил по 7 бочек, в которых содержится 3,5 бочки вина.

Из 7 бочек:

- a полных
- b полных наполовину
- c пустых

Значения a и b заключены в диапазоне 0..7, причём их общее число не может превышать 7.

На долю пустых бочек c приходится 7 – a – b.

Решение задачи мы начнём с составления списка всех возможных распределений 7 бочек разной наполненности между грубыми русскими мужиками:

```
uses
   System;

// Фольклор Д31

begin
   // заголовок окна:
   Console.Title := 'Математический фольклор. Задача Д31';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Проспись и пей!');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.

type
   aoi = array of integer;

// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   // список бочек у каждого:
   var lst := new List<aoi>();

   // полные бочки:
   for var a := 0 to 7 do
   begin
      // бочки, наполненные наполовину:
      for var b := 0 to 7 do
      begin
         // их должно быть не больше 7:
         if (a + b > 7) then
            continue;
```

Тут же мы должны учесть условие, что каждый из них получил по 3,5 бочки вина. Чтобы упростить поверку и избежать дробей, умножим число бочек и полученный объём вина на 2:

```
// должно быть не более 7/2 бочек вина:
if (2 * a + b <> 7) then
   continue;
```

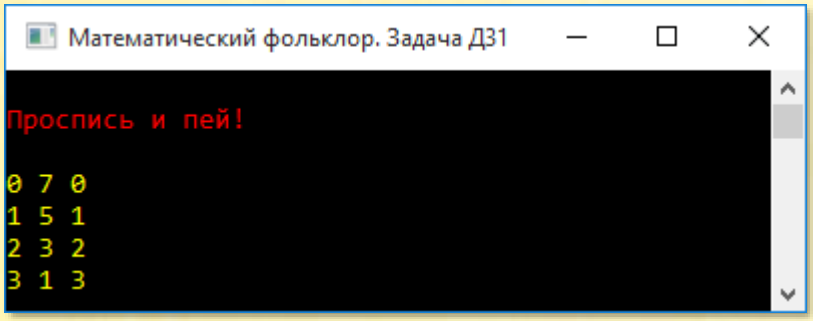
Пополняем список новой комбинацией:

```
        // записываем комбинацию:
        //var li := new List<integer>();
        //li.AddRange(Seq(a, b, 7 - a - b ));
        //lst.Add(li);
        lst.Add(new integer[](a, b, 7 - a - b));
    end
  end;
```

Давайте из любопытства краем глаза взглянем на этот бочковый список:

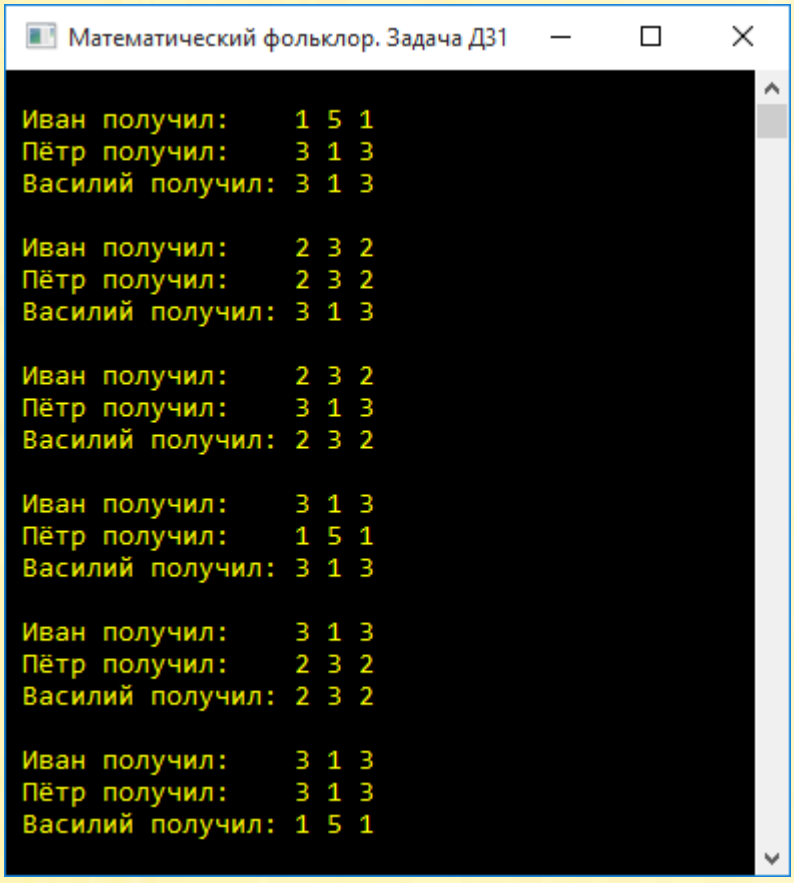
```
foreach var c in lst do
  begin
    foreach var i in c do
       Console.Write(i + ' ');
    Console.WriteLine();
  end;
  Console.WriteLine();
```

Как нам показывает рисунок, всего существует 4 способа поделить бочки по-мужски, то есть по-честному.



Теперь нужно так скомбинировать эти способы, чтобы число разнозаполненных бочек равнялось семи:

```
foreach (var i in lst)
  foreach var i in lst do
      foreach var p in lst do
         foreach var v in lst do
         begin
            for var n := 0 to 3 - 1 do
               if (i[n] + p[n] + v[n] = 7) then
               begin
                  Console.WriteLine(' Иван получил:    ' + i[0] +
                                    ' ' + i[1] + ' ' + i[2]);
                  Console.WriteLine(' Пётр получил:    ' + p[0] +
                                    ' ' + p[1] + ' ' + p[2]);
                  Console.WriteLine(' Василий получил: ' + v[0] +
                                    ' ' + v[1] + ' ' + v[2]);
                  Console.WriteLine();
                  break;
               end;
         end;
   Console.WriteLine();
end;
```



Всего имеется 6 способов дележа бочек между мужиками:

В задаче неявно предполагается, что **вино нельзя переливать из одной бочки в другую**.

В книге *Математический фольклор*, в *Задаче 43* это ограничение отсутствует:

> *Отец имел 8 полных, 8 полупустых и 8 пустых бочек.*
>
> ***Может ли он разделить их между тремя сыновьями так, чтобы они получили по одинаковому количеству полных, полупустых и пустых бочек?***

Совершенно очевидно, что 8 бочек невозможно поровну разделить на 3 части. Но мы можем из двух полных бочек **перелить** вино в в другие бочки, тогда полных бочек останется 6, и мы сможем передать каждому сыну по 2 полные бочки.

Переливать вино в полупустые бочки смысла нет, поэтому из двух полных бочек мы переливаем вино в 4 пустые бочки, превращая их в полупустые. Теперь мы имеем 12 полупустых бочек. От 8 пустых бочек осталось 4, но к ним добавились 2 бочки, которые раньше были полными. В результате этих переливаний из полного в порожнее мы получили другой расклад бочек:

6 12 6

Число всех видов бочек кратно 3, так что мы легко поделим их между сыновьями поровну:

2 4 2

Каждому досталось по 2 полные и 2 пустые бочки, а также по 4 полупустые бочки.

Чешские сливы

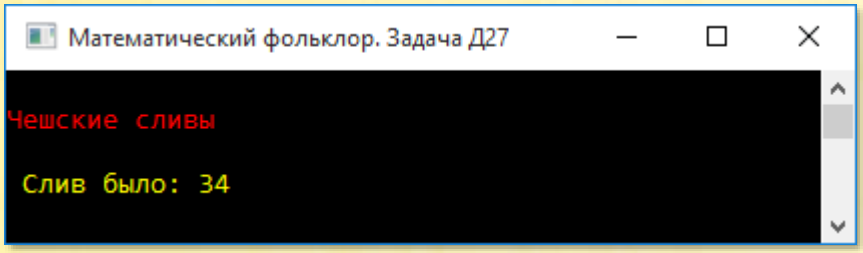
Во дворец чешской королевы Любуши пришли трое просить руки её дочери. И она им предложила сначала ответить на такой вопрос:

- В корзине находятся сливы. Если первому из вас дать половину всех слив и ещё одну, второму – половину от оставшихся и ещё две сливы, третьему – половину от нового остатка и ещё три сливы, то слив в корзине не останется. Продолжать разговор я буду только с тем, кто ответит, сколько слив было в корзине.

Что должны ответить кандидаты?

Легко увидеть в этой задаче усложнённый вариант *Русских яблок*, поэтому в сжатые сроки мы переделываем исходный код и получаем достойный ответ:

```
uses
   System;

// Фольклор Д27

// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
    Console.ForegroundColor := ConsoleColor.Yellow;

    for var sliva := 1 to integer.MaxValue do
    begin
        var ost := double(sliva);
        for var i := 1 to 3 do
        begin
            ost := ost/2 - i;
        end;
        if (ost = 0) then
        begin
            Console.WriteLine(' Слив было: ' + sliva);
            Console.WriteLine();
            exit;
        end
    end
end;

begin
   // заголовок окна:
   Console.Title := 'Математический фольклор. Задача Д27';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Чешские сливы');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```



Французский покупатель

Пьер был в хорошем настроении, имел некоторую сумму денег и решил пойти в магазин. У хозяина магазина он занял столько денег, сколько у него было, после чего потратил 1 франк. Затем он зашёл во второй магазин, снова занял у хозяина, сколько имел, и потратил 2 франка. Он посетил ещё два магазина, где занимал столько, сколько имел, и тратил соответственно 5 и 6 франков. Когда он вышел из последнего магазина, то оказалось, что денег у него нет.

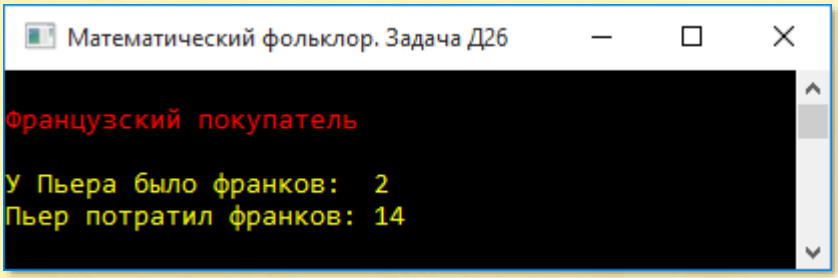
Сколько денег было у Пьера первоначально и сколько он их всего потратил?

Если бы Пьер был осмотрительнее и не занимал деньги, то задача решалась бы точно так же, как с яблоками и сливами. Здесь же мы должны учесть приток капитала и нелинейные расходы Пьера в магазинах:

```
uses
   System;

// Фольклор Д26


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
    Console.ForegroundColor := ConsoleColor.Yellow;

    var traty := new integer[] ( 0, 1, 2, 5, 6 );
    for var money := 1 to integer.MaxValue do
    begin
        var sum := money;
        for var n := 1 to 4 do
        begin
            sum := sum + sum - traty[n];
        end;
        if (sum = 0) then
        begin
            Console.WriteLine('У Пьера было франков:  ' + money);
            Console.WriteLine('Пьер потратил франков: ' +
                              traty.Sum());
            Console.WriteLine();
            break;
        end
    end
end;


begin
   // заголовок окна:
   Console.Title := 'Математический фольклор. Задача Д26';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Французский покупатель');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();
```

```
    Solve();

    Console.WriteLine();
    Console.ForegroundColor := ConsoleColor.Red;
end.
```

Ответ на задачу весьма забавный: у Пьера было 2 франка, а потратил он 14!



Болгарский парикмахер

Трое мужчин пришли к парикмахеру. Побрив первого, парикмахер сказал:

- Посмотри, сколько денег в ящике стола и возьми 2 лева сдачи.

То же сказал парикмахер и второму, и третьему. После того как трое ушли, оказалось, что в кассе нет денег.

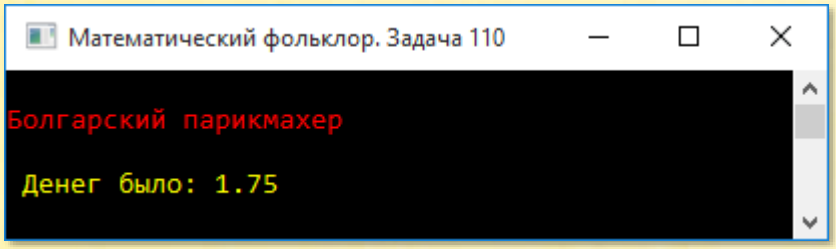
Сколько денег было в кассе перед тем, как заплатил первый мужчина?

В этой задаче расчёты лучше вести не в левах, а в стотинках, чтобы избежать неточных вычислений с данными типа *double*. А окончательный результат мы переведём в левы, как в книжном ответе:

```
uses
   System;

// Фольклор 110


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
    Console.ForegroundColor := ConsoleColor.Yellow;

    for var money := 100 to integer.MaxValue do
    begin
        var ost := money;
        for var i := 1 to 3 do
        begin
            ost := ost + ost - 200;
        end;
        if (ost = 0) then
        begin
            Console.WriteLine(' Денег было: ' + money/100.0);
            Console.WriteLine();
            break;
        end
    end
end;


begin
   // заголовок окна:
   Console.Title := 'Математический фольклор. Задача 110';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Болгарский парикмахер');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();
```

```
    Console.WriteLine();
    Console.ForegroundColor := ConsoleColor.Red;
end.
```

Других сложностей задача нам не предлагает. А **ответ** такой: в кассе было 1,75 лева:



Болгарские сливы

Сливы – они и есть сливы: что чешские, что болгарские, поэтому мы легко справимся и с этой задачей!

Из корзины слив одна женщина взяла половину и ещё одну, другая женщина взяла половину от оставшихся и ещё одну, третья женщина взяла половину от последнего остатка и ещё 3 сливы, после чего в корзине слив не осталось.

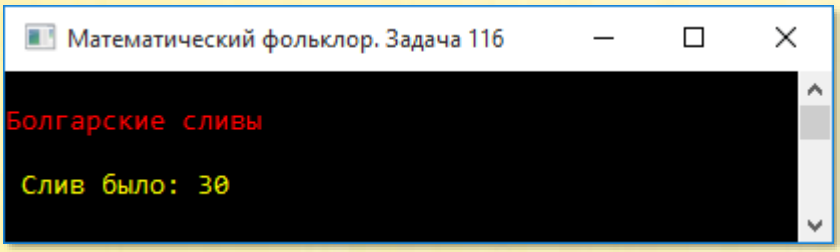
Сколько слив было сначала в корзине?

Но нужно принять во внимание, что женщины брали нерегулярное число слив. Как обычно, для хранения слив и прочих фруктов мы используем массив:

```
uses
   System;

// Фольклор 116


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
    Console.ForegroundColor := ConsoleColor.Yellow;

    var brali := new integer[] ( 0, 1, 1, 3 );
    for var sliva := 1 to integer.MaxValue do
    begin
        var ost := double(sliva);
        for var i := 1 to 3 do
        begin
            ost := ost / 2 - brali[i];
        end;
        if (ost = 0) then
        begin
            Console.WriteLine(' Слив было: ' + sliva);
            Console.WriteLine();
            exit;
        end
    end
end;


begin
   // заголовок окна:
   Console.Title := 'Математический фольклор. Задача 116';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Болгарские сливы');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
```

```
    Console.ForegroundColor := ConsoleColor.Red;
end.
```

А было в корзине 30 слив:



Немецкий вопрос

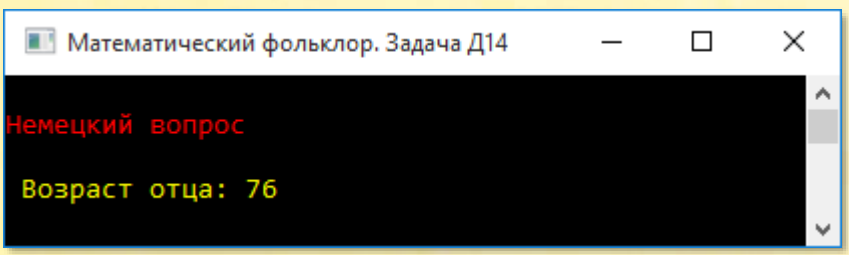
Сын спросил отца, сколько ему лет. Отец ответил так:

- Если к моим годам прибавить их половину, затем их четверть и ещё один год, то получится 134 года.

Сколько лет отцу?

Нетрудно заметить, что возраст отца кратен 4. Перебираем все числа, делящиеся на 4 без остатка, в бесконечном цикле ***while*** до тех пор, пока не выполнится условие задачи:

```
uses
   System;

// Фольклор Д14

// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
    Console.ForegroundColor := ConsoleColor.Yellow;
var let := 0;
while(true) do
    begin
        if (let + let div 2 + let div 4 + 1 = 134) then
        begin
            Console.WriteLine(' Возраст отца: ' + let);
            Console.WriteLine();
            break;
        end;
        let += 4;
    end
end;

begin
   // заголовок окна:
   Console.Title := 'Математический фольклор. Задача Д14';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Немецкий вопрос');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```



Наверняка у этого отца взрослый сын. Странно, что он не знает его возраста...

Индийские рупии

Один говорит другому:
- Дай мне 100 рупий, и я буду в 2 раза богаче тебя.

Второй отвечает:
- А если ты дашь мне 10 рупий, то я стану в 6 раз богаче тебя.

Сколько денег было у каждого?

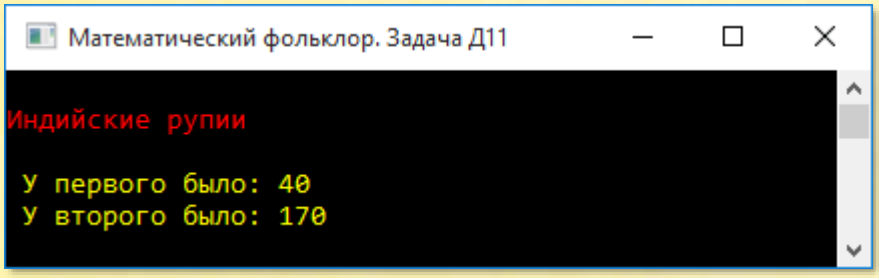
Очевидно, что у **первого** было не меньше 10 рупий, а у **второго** – не меньше 100. Верхний предел нам неизвестен, но нельзя организовать 2 бесконечных вложенных цикла, поэтому во втором (но не в первом!) цикле мы полагаем, что у второго было не больше 1000 рупий. В случае неудачи мы повысим его денежное содержание.

По условию задачи, должны выполниться 2 условия, которые легко изложить на языке *паскаль*:

```
uses
   System;

// Фольклор Д11


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
```

```
   Console.ForegroundColor := ConsoleColor.Yellow;

   for var one := 10 to integer.MaxValue do
      for var two := 100 to 1000 do
      begin
         var uslovie1 := one + 100 = 2 * (two - 100);
         var uslovie2 := two + 10 = 6 * (one - 10);
         if (uslovie1 and uslovie2) then
         begin
            Console.WriteLine(' У первого было: ' + one);
            Console.WriteLine(' У второго было: ' + two);
            Console.WriteLine();
            exit;
         end
      end
end;


begin
   // заголовок окна:
   Console.Title := 'Математический фольклор. Задача Д11';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Индийские рупии');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

Итак, у первого было 40 рупий, а у второго – 170:



Русские гуси

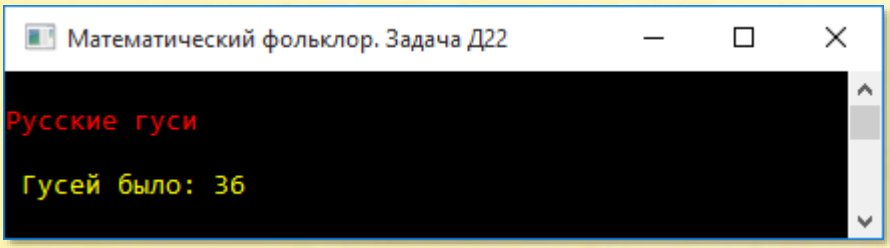
Очень популярная задача про стаю гусей:

> *Летела стая гусей, а навстречу им летит один гусь и говорит:*
>
> *- Здравствуйте, сто гусей!*
>
> *Старший гусь, их вожак, ответил:*
>
> *- Нас не сто гусей. Но если взять сколько есть, да ещё столько, да ещё пол-столька, да четверть столька, да ещё вместе с тобой, нас будет 100.*
>
> ***Сколько было гусей?***

Чтобы решить задачу, нужно просто внимательно прочитать её и записать условие на математическом языке:

```
uses
   System;

// Фольклор Д22


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
```

```
   Console.ForegroundColor := ConsoleColor.Yellow;

   for var gusey := 1 to integer.MaxValue do
      if (gusey + gusey + gusey div 2 + gusey div 4 + 1 = 100)
then
      begin
         Console.WriteLine(' Гусей было: ' + gusey);
         Console.WriteLine();
         break;
      end;
end;


begin
   // заголовок окна:
   Console.Title := 'Математический фольклор. Задача Д22';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Русские гуси');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```



У болгар имеется подобная задача про аистов (*Математический фольклор, Задача 124*):

> *Шёл человек с поля, а навстречу ему аисты.*

- Здравствуйте, сто аистов, - поздоровался он.

А старший аист, их вожак, ответил:

- Нас не сто! А если к нам подлетит ещё столько, сколько нас, и ещё половина, и ещё четверть, и вместе с тобой нас станет 100!

Сколько было аистов?

В XVIII веке эта задача была популярна не только в России, но и в Германии, только место гусей в ней заняли голуби (*Математический фольклор, Задача Д21*):

Крестьянин шёл с поля, встретил стаю голубей и сказал:

- Добрый день, сто голубей.

Но один из голубей ответил:

- Нет, нас не сто! Но если взять ещё столько, ещё половину, ещё четверть, и вместе с тобой нас будет сто.

Сколько было голубей?

И неожиданная российская интерпретация этой же задачи (*Математический фольклор, Задача Д20*):

Отец спросил учителя своего сына, сколько у него учеников. Учитель ответил:

- Если взять ещё столько учеников, сколько уже есть, да ещё половину и четверть их, да ещё другого твоего сына, то станет точно 100.

Сколько у него учеников?

Насос Эдисона

Эта задача основана на историческом анекдоте. Известно, что калитки с насосом у Эдисона не было, да и быть не могло, поскольку её никто не смог бы открыть, и Эдисон быстро остался бы без друзей.

Томас Альва Эдисон обладал тонким чувством юмора. Его многочисленные посетители часто удивлялись, почему калитка в саду перед домом великого изобретателя открывается с трудом. Наконец, один из друзей спросил у Эдисона:

- Неужели такой технический гений, как ты, не может отрегулировать какую-то калитку?

- Калитка отрегулирована именно так, как надо, - смеясь возразил Эдисон. – Я сделал от неё привод к цистерне, и каждый, кто приходит ко мне, накачивает в цистерну 20 литров воды.

Если бы каждый посетитель вместо 20 литров накачивал в цистерну 25 литров воды, то для заполнения цистерны понадобилось бы на 12 человек меньше.

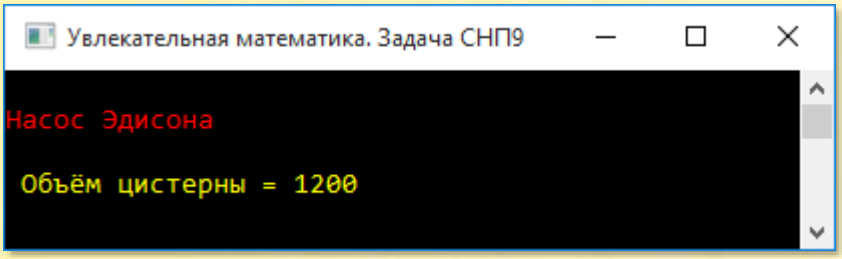
Сколько воды вмещает цистерна?

Задачу Эдисона решить гораздо легче, чем отворить его калитку. Достаточно переписать условие задачи на *паскаль*. При этом мы должны учесть, что в нём присутствует операция деления, поэтому мы выбираем **вещественный** тип для переменной **vol**:

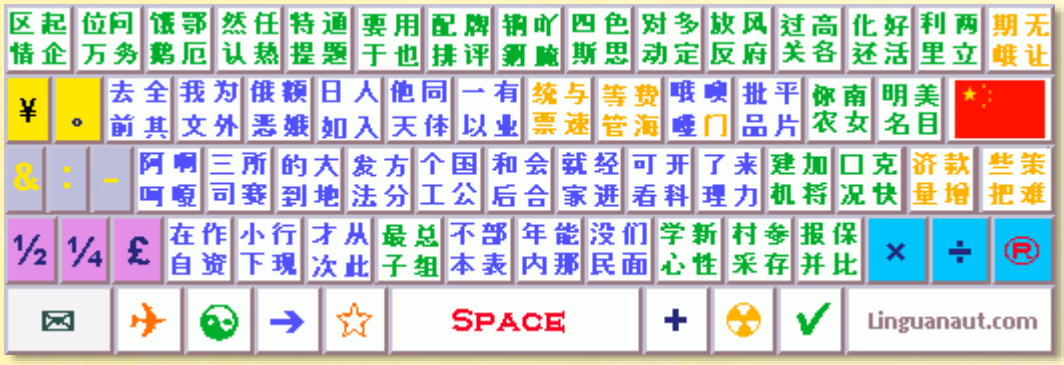
```
uses
   System;

// Увлекательная математика СНП9


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   for var v := 0 to integer.MaxValue do
   begin
      var vol := double(v);
      if (vol / 25 = vol / 20 - 12) then
      begin
         Console.WriteLine(' Объём цистерны = ' + vol);
         Console.WriteLine();
         break;
      end
   end
end;


begin
   // заголовок окна:
   Console.Title := 'Увлекательная математика. Задача СНП9';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Насос Эдисона');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();
```

```
    Console.WriteLine();
    Console.ForegroundColor := ConsoleColor.Red;
end.
```

У большого мастера и цистерна была соответствующего объёма!



Китайская арифметика



Всем известно выражение *китайская грамота*, означающее сложные и непонятные вещи. Напротив, китайская арифметика и проста, и понятна. Вот наглядный пример из трактата *Начала искусства вычисления*, написанного в 1593 году:

> *Найти число, которое при делении на 3 даёт в остатке 2, при делении на 5 даёт в остатке 3 и при делении на 7 – снова 2.*

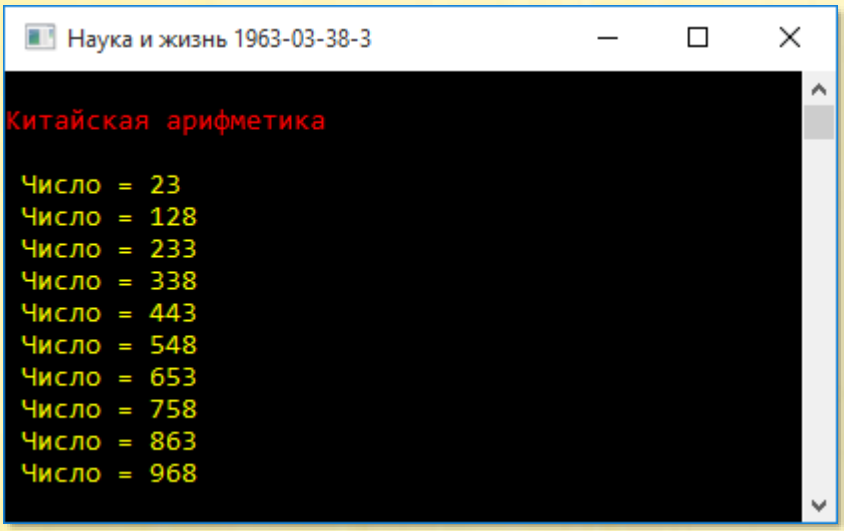
Мы перебираем целые неотрицательные числа в «бесконечном» цикле ***for***, пока не найдём требуемое число:

```
uses
   System;

// Наука и жизнь 1963-03-38-3


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
    Console.ForegroundColor := ConsoleColor.Yellow;

    var variant := 0;
    for var num := 0 to integer.MaxValue do
    begin
        if ((num mod 3 = 2) and (num mod 5 = 3) and (num mod 7 = 2))
then
        begin
            Console.WriteLine(' Число = ' + num);
            variant += 1;
            if (variant = 10) then
                break;
        end
    end;
    Console.WriteLine();
end;


begin
   // заголовок окна:
   Console.Title := 'Наука и жизнь 1963-03-38-3';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Китайская арифметика');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

Минимальное число равно 23, а остальные можно найти по формуле:

Китайское число = 23 + 105n,

где n – целое неотрицательное число.



Задача Этьена Безу

Этьен Безу – французский математик 18 века - преподавал математику в Училище гардемаринов и в Королевском артиллерийском корпусе.

Нам известна такая задача Этьена Безу:

По контракту работникам причитается по 48 франков за каждый отработанный день, а за каждый неотработанный день с них взыскивается по 12 франков. Через 30 дней выяснилось, что работникам ничего не причитается.

Сколько дней они отработали в течение этих 30 дней?

Пусть они отработали *x* дней.

За эти дни им причитается по 48x франков.

Остальные (30 - x) дней были «выходными».

За них работники должны вернуть 12(30 - x) франков.

Всего за 30 дней каждому работнику причитается 0 франков:

$$48x - 12(30 - x) = 0$$

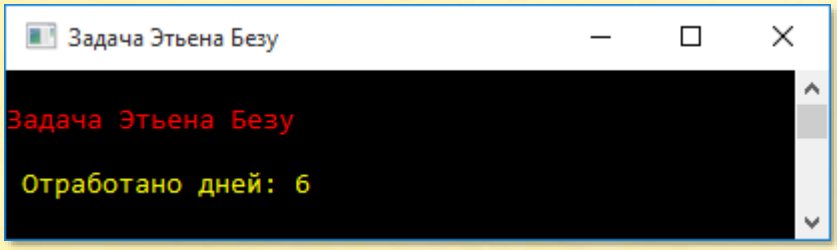
Так как число дней – **целое**, то уравнение решается простым перебором в «бесконечном» цикле *for*. Когда условие задачи выполнится, цикл прекращается.

Вместо математической переменной *x* мы создадим программистскую переменную **days**. Вот и вся задача:

```
uses
   System;

// Задача Этьена Безу


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   for var days := 0 to integer.MaxValue do
   begin
      if (48 * days - 12 * (30 - days) = 0) then
      begin
         Console.WriteLine(' Отработано дней: ' + days);
         break;
      end
   end;
   Console.WriteLine();
end;
```

```
begin
   // заголовок окна:
   Console.Title := 'Задача Этьена Безу';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Задача Этьена Безу');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

А ответ такой:

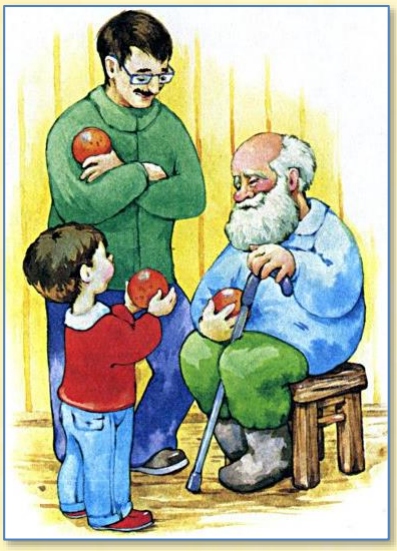


Задачу легко решить в уме.

Работники за 1 отработанный день получают в 4 раза больше, чем с них взыскивают за каждый неотработанный день. В итоге они не заработали ничего. Это значит, что они отработали в 4 раза меньше дней, чем «прогуляли». Это составляет одну пятую от 30 дней, то есть 6 дней.

Кому сколько лет?

Задача 70 из книги *Русские головоломки*:



Дед, отец и сын во время прогулки встретили знакомого, который спросил их, сколько им лет.
- Нам 121 год, - ответил за всех дед и важно зашагал вперёд.
Тогда знакомый, продолжая интересоваться их возрастом, спросил отца:
- Ну, скажите же, сколько вам лет?
- Мне вместе с сыном 44 года, - отвечал отец, - а сын на 28 лет моложе меня.
Так знакомому и не пришлось узнать, сколько лет каждому из них. Не сообразите ли вы?

Легко сообразить, что **деду** 121 – 44 = **77** лет, потому что остальные 44 года из общей суммы приходятся на отца и сына.

Пусть отцу *x* лет, а сыну – *y*. Вместе им 44 года:

$x + y = 44$

Условие, что сын на 28 лет моложе отца, даёт нам второе уравнение:

$y = x - 28$

Заменив математические иксы-игреки более осмысленными переменными **otets** и **syn**, мы быстро составляем **программу**:

```
uses
   System;

// Кому сколько лет


// РЕШАЕМ ЗАДАЧУ
procedure Solve();
```

```
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   for var syn := 1 to integer.MaxValue do
   begin
      var otets := 28 + syn;
      if (otets + syn = 44) then
      begin
         Console.WriteLine(' Возраст отца:  ' + otets);
         Console.WriteLine(' озраст сына:   ' + syn);
         break;
      end
   end;
   Console.WriteLine();
end;

begin
   // заголовок окна:
   Console.Title := 'Кому сколько лет';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('ЗКому сколько лет');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

Ответ не заставил себя долго ждать:

Как-то даже неудобно перед компьютером, что мы задаём ему такие простые задачки...

Ноги и головы

Задача 71 из книги *Русские головоломки*:

На скотном дворе гуляли гуси и поросята. Хозяин и его сын вышли на двор, посмотрели на живность и пошли в поле. По дороге сын и спрашивает:

- Отец, сколько у нас на скотном дворе гусей и сколько поросят?

- А вот угадай-ка сам, - ответил отец. – *Если считать по головам, то на дворе 25 голов, а если по ногам, то 70 ног.*

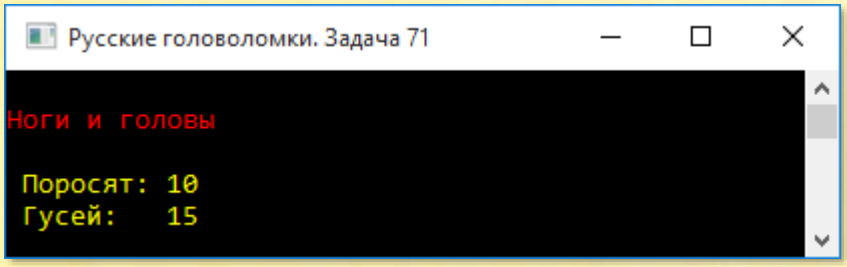
Сколько было гусей и сколько поросят?

Эта задача очень похожа на кроличье-фазанью, которых здесь заменили поросята и гуси:

```
uses
   System;

// Ноги и головы

const
   // общее число голов:
   HEADS = 25;
   // общее число ног:
   LEGS = 70;

// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   // макс. число поросят:
```

```
    var maxPoros := LEGS div 4;

    // решаем задачу:
    for var i := 0 to maxPoros do
    begin
              //число гусей:
       var nGus := HEADS - i;

       if (i * 4 + nGus * 2 = LEGS) then
       begin
          Console.WriteLine(' Поросят: ' + i);
          Console.WriteLine(' Гусей:   ' + nGus);
       end
    end;
    Console.WriteLine();
end;

begin
    // заголовок окна:
    Console.Title := 'Русские головоломки. Задача 71';
    Console.WriteLine('');
    Console.ForegroundColor := ConsoleColor.Red;
    Console.WriteLine('Ноги и головы');
    Console.ForegroundColor := ConsoleColor.Green;
    Console.WriteLine();

    Solve();

    Console.WriteLine();
    Console.ForegroundColor := ConsoleColor.Red;
end.
```

Задача простейшая даже для компьютера. Инвентаризация закончена мгновенно: гусей было 15 голов, а поросят – 10:



Задания для самостоятельного решения

Задача *Фольклор Д43*

Бразильский кофе

Каждое утро Рейнальдо выпивает 1 чашку крепкого кофе, которая содержит определённую дозу кофеина. Однажды ему предстояло приготовить себе традиционный кофе из партии зёрен, из которых уже было извлечено 97% кофеина.

Сколько чашек такого кофе должен выпить Рейнальдо, чтобы в них содержалось столько кофеина, сколько содержалось его в одной чашке крепкого кофе?

Простая задача на пропорции.

Ответ: 33 ⅓ чашки.

Задача *Фольклор Д29*

Возраст сыновей

Когда у отца спросили, сколько лет его сыновьям, он ответил:

- Старшему в три раза больше, чем младшему, а им вместе столько лет, сколько было мне 29 лет назад. Сейчас мне 45 лет.

Сколько лет сыновьям?

---

29 лет назад отцу было 16 лет. Отсюда следует...

Ответ: 12 лет и 4 года.

Задача *Фольклор Д25*

Который час?

На вопрос: «Сколько времени?» - был дан такой ответ:

- Две пятых времени, прошедшего от полночи до этого момента, равно двум третьим времени, которое осталось до полудня.

Сколько сейчас времени?

От полночи до полудня 12 часов. Если от полночи прошло *x* часов, то до полудня осталось *(12-x)* часов. Осталось составить простейшее уравнение с одним неизвестным.

Ответ: 7 часов 30 минут.

Задача *Фольклор Д24*

Австралийский скотовод

Скотовод завещал трём своим сыновьям – Альфреду, Джону и Чарльзу – разделить стадо овец следующим образом: Альфред получит на 20% больше Джона и на 25% больше Чарльза. Часть Джона – 3600 овец.

Сколько овец получит Чарльз?

Простая задача, решение которой не требует дополнительных пояснений.

Ответ: 3456.

Задача *Фольклор Д23*

Французские братья

Три брата хотели купить дом, который стоил 26000 франков. Условились, что первый даст половину, второй – треть, а третий – четверть стоимости.

Сколько денег даст каждый?

Легко посчитать, что ½ + ⅓ + ¼ = 13/12, что соответствует 26000 франков. Теперь найти долю каждого брата не составит труда.

Ответ: 12000, 8000 и 6000.

СОВРЕМЕННЫЕ ЗАДАЧИ

Современных занимательных задач и не сосчитаешь! Их можно найти и в книгах, и в журналах, и в газетах, и в Интернете. Фантазии авторов нет предела. Нет ни одного раздела математики, который не попал бы в поле зрения составителей головоломных задач.

В этой главе вы снова встретитесь с задачами из разных стран мира. Решайте их – и от математики можно получать удовольствие. Да ещё какое!

Кузнечики

В седьмом номере журнала *Квантик* за 2016 год, на странице 32 напечатана вот такая **конкурсная задача**:

> **31.** Двадцать пять ребят пошли в лес и стали ловить кузнечиков. Несколько ребят поймали по одному кузнечику, половина оставшихся ребят поймали по два кузнечика, а остальные не смогли поймать ни одного. Сколько всего кузнечиков поймали ребята?

Задача скорее на сообразительность, чем математическая. Такие задачи очень простые, да вот только догадаться сложно, как же их решать.

Можно решить задачу **алгебраически**, обозначив буквой *x* число ребят, которые поймали по одному кузнечику. Но на *паскале* задачу лучше решать простым **перебором**.

Мы знаем, что *x* не меньше 1 и не больше 25. Причём это число **нечётное**, иначе число оставшихся ребят также будет нечётным, а от такого числа нельзя взять **ровно половину**.

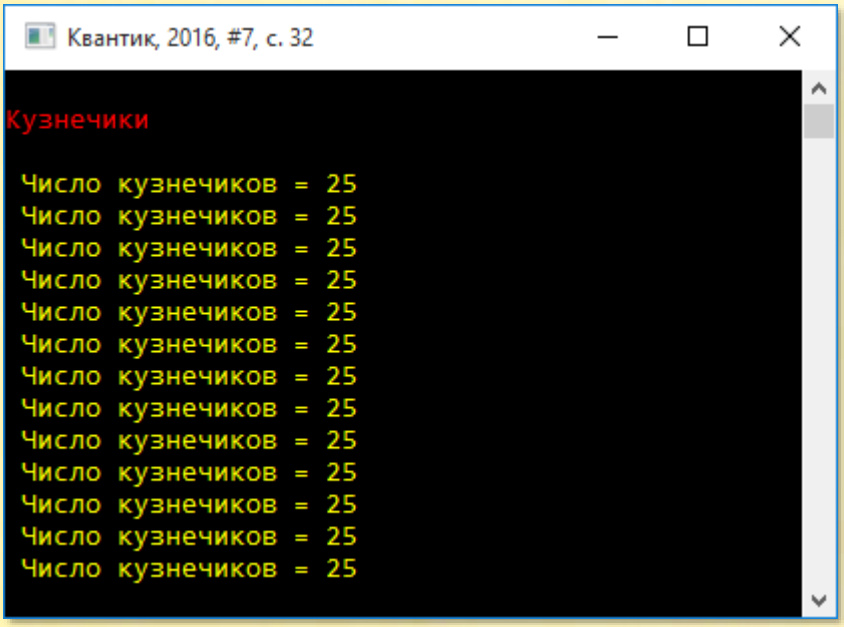
Создаём переменную **x**. Её начальное значение равно **1**. В цикле ***while*** мы изменяем значение переменной *x* на 2, чтобы оно всегда было нечётным:

```
uses
   System;

// Кузнечики

// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;
```

```
    var x := 1;
    while(x <= 25) do
    begin
       var nOstReb := 25 - x;
       var nKuzn := x + (25 - x) / 2 * 2;
       Console.WriteLine(' Число кузнечиков = ' + nKuzn);
       x += 2;
    end;
    Console.WriteLine();
end;


begin
    // заголовок окна:
    Console.Title := 'Квантик, 2016, #7, с. 32';
    Console.WriteLine('');
    Console.ForegroundColor := ConsoleColor.Red;
    Console.WriteLine('Кузнечики');
    Console.ForegroundColor := ConsoleColor.Green;
    Console.WriteLine();

    Solve();

    Console.WriteLine();
    Console.ForegroundColor := ConsoleColor.Red;
end.
```

Запускаем программу и смотрим на список:

Оказывается, каким бы ни было число ребят, поймавших по 1 кузнечику, общее число кузнечиков всегда равняется 25.



А теперь легко догадаться, почему так происходит. Выражение

$x + (25 - x) / 2 * 2$

при любом значении ***x*** равно 25:

$x + (25 - x) = 25$

Задача пустяковая, а ведь не сразу это поймёшь!

Бельгийские числа

Типичная **переборная задача** на поиск чисел, цифры которых удовлетворяют некоторому условию:

> *Найти трёхзначные числа вида abc, цифры которых удовлетворяют уравнению:*
>
> $a^2 - b^2 - c^2 = a - b - c$
>
> *Все 3 цифры числа должны быть различны.*

Как обычно в таких случаях, выписываем 3 вложенных цикла, в которых и перебираем все возможные комбинации цифр:

```
uses
   System, System.Linq;

// Увлекательная математика 12

begin
   // заголовок окна:
   Console.Title := 'Увлекательная математика. Задача 12';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Бельгийские числа');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

Для разнообразия поместим все цифры (однозначные числа) в коллекцию digs, откуда вынимать их легко и приятно:

```
// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   //var digs := Enumerable.Range(0, 10);
   var digs := Range(0, 10);
```

Так как числа трёхзначные, то первая цифра не может быть нулём:

```
   foreach var d1 in digs do
   begin
      if (d1 = 0) then
         continue;
```

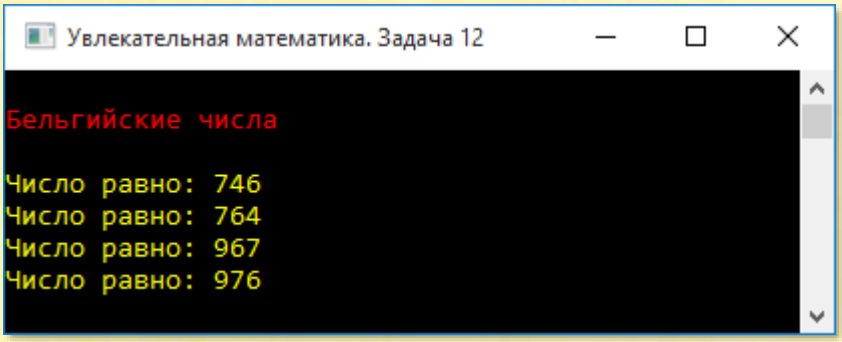
А вторая и третья должны отличаться друг от друга и от первой цифры:

```
foreach var d2 in digs do
begin
   if (d2 = d1) then
      continue;
   foreach var d3 in digs do
   begin
      if ((d3 = d2) or (d3 = d1)) then
         continue;
```

Когда условие задачи выполнится, мы напечатаем очередное решение:

```
            if (d1 * d1 - d2 * d2 - d3 * d3 = d1 - d2 - d3) then
            begin
               Console.WriteLine('Число равно: ' + (d1 * 100 +
                                 d2 * 10 + d3));
            end
         end
      end
   end;
   Console.WriteLine();
end;
```

На рисунке вы видите все четыре решения этой бесхитростной задачки:



Немецкая копилка

Как говорится, опять за рыбу деньги!

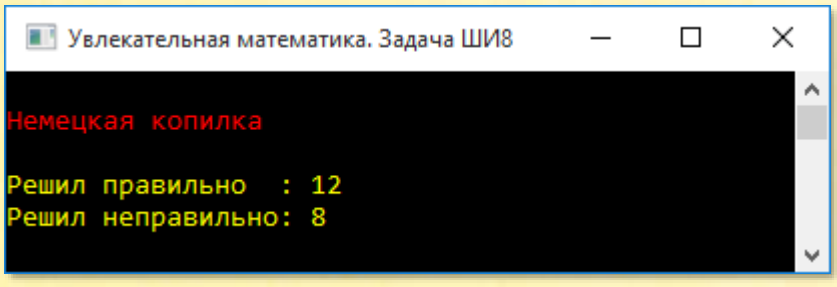
> *Отец обещал сыну за каждую правильно решённую задачу опускать в копилку по 10 пфеннигов. За каждую неправильно решённую задачу сын должен возвращать отцу по 5 пфеннигов. После того как было решено 20 задач, у сына в копилке оказалось 80 пфеннигов.*
>
> *Сколько задач сын решил неправильно и сколько без единой ошибки?*

Задача очень простая, поэтому не будем тратить компьютерное и собственное время на излишние комментарии:

```
uses
   System;

type
   int = integer;
   bool = boolean;

// Увлекательная математика ШИ8

// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   for var verno := 0 to 20 do
   begin
```

```
      if (verno * 10 - (20 - verno) * 5 = 80) then
      begin
         Console.WriteLine('Решил правильно  : ' + verno);
         Console.WriteLine('Решил неправильно: ' + (20 - verno));
         Console.WriteLine();
      end
   end
end;


begin
   // заголовок окна:
   Console.Title := 'Увлекательная математика. Задача ШИ8';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Немецкая копилка');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

Конечно, 80 пфеннигов деньги небольшие, но, судя по ответу, сынок и на них не наработал!



Ошибки

Микрокалькулятор с ЕГГОГом

И ещё одна задачка из школьной жизни:

> *Марина сделала в диктанте несколько ошибок. Гриша у неё всё списал да ещё допустил 5 ошибок.*
>
> ***Сколько ошибок допустил каждый**, если учитель обнаружил в двух диктантах 35 ошибок?*

Объявим переменную **error** для подсчёта числа ошибок и инициализируем её нулём:

```
uses
   System;

type
   int = integer;
   bool = boolean;

// Математический фольклор. Задача Д1

// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   var error := 0;
```

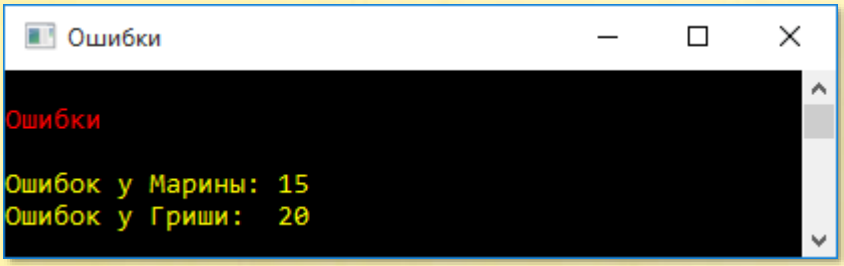
В старых микрокалькуляторах английское слово **ERROR** (*ошибка*) на экране выглядело, как русское, но совершенно непонятное слово **ЕГГОГ**.

Будем добавлять по одной ошибке в цикле ***reprat-until*** до тех пор, пока не выполнится условие задачи, то есть число ошибок у Марины и у Гриши вместе взятых не достигнет 35:

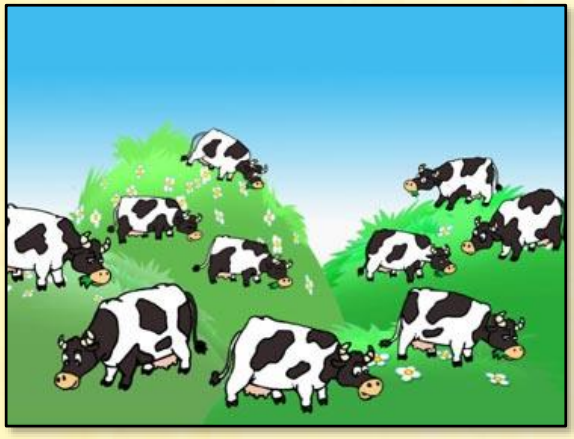
```
  repeat
      error += 1;
  until (error + (error + 5) = 35);
```

Когда это событие произойдёт, мы напечатаем ответ на экране:

```
    Console.WriteLine('Ошибок у Марины: ' + error);
    Console.WriteLine('Ошибок у Гриши:  ' + (error + 5));
    Console.WriteLine();
end;

begin
    // заголовок окна:
    Console.Title := 'Ошибки';
    Console.WriteLine('');
    Console.ForegroundColor := ConsoleColor.Red;
    Console.WriteLine('Ошибки');
    Console.ForegroundColor := ConsoleColor.Green;
    Console.WriteLine();

    Solve();

    Console.WriteLine();
    Console.ForegroundColor := ConsoleColor.Red;
end.
```



Коровы



А теперь решим задачку из сельской жизни:

> *На лугу паслось несколько коров. У них ног на 24 больше, чем голов.*
>
> *Сколько коров паслось на лугу?*

Ответ на задачу мы найдём в условии задачи: коров паслось несколько. Это, конечно, шутка, а задачка-то ведь нешуточная!

Обозначим число коров через k:

```
uses
   System;

type
   int = integer;
   bool = boolean;

// Коровы

// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;
```

```
    //число коров:
    var k := 0;
```

Будем добавлять по одной корове до тех пор, пока не выполнится условие задачи, то есть число ног у коров за вычетом числа самих коров не станет равно 24:

```
  repeat
      k += 1
  until (k * 4 - k = 24);

   Console.WriteLine('Коров: ' + k);
   Console.WriteLine();
end;


begin
   // заголовок окна:
   Console.Title := 'Коровы';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Коровы');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

Ответ: на лугу паслось 8 коров:.

Мешки с сокровищами

А вот задача из книги Катта Мурти (Katta G. Murty) *Junior Level Web-Book Optimization Models for decision Making*, Chapter 7:

> *7.8.2: A merchant has bags of emeralds (nine to a bag), and rubies (four to a bag). He has a total of 59 jewels. Required to find how many of his jewels are rubies.*

Её можно перевести так:

> *У торговца есть сумки: с изумрудами (по 9 штук в сумке) и с рубинами (по 4 штуки в сумке). В общей сложности у него 59 драгоценностей.*
>
> ***Требуется найти, сколько из них - рубины.***

Для общего числа драгоценностей мы введём константу TOTAL:

```
uses
   System;

// Мешки с сокровищами

const
   // общее число драгоценностей:
   TOTAL = 59;
```

Число сумок с драгоценностями может изменяться от нуля до какого-то предельного значения, которое легко вычислить простым делением общего числа драгоценностей *TOTAL* на число драгоценностей в каждой сумке. Как вы помните, это 9 для изумрудов (*Emeralds*) и 4 – для рубинов (*Rubies*):

```
// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   // макс. число сумок с изумрудами:
   var maxEmeralds := TOTAL div 9;
   // макс. число сумок с рубинами:
   var maxRubies := TOTAL div 4;
```

С тем же успехом для этих целей можно использовать не переменные, а **константы**.

Теперь мы можем во вложенных циклах ***for*** перебрать все варианты числа сумок:

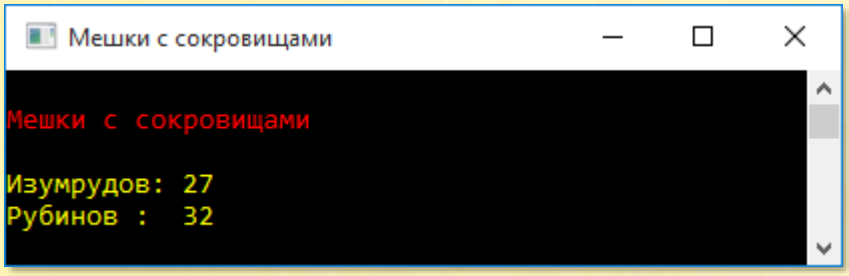
- с изумрудами – переменная цикла j - от *0* до *maxEmeralds*
- с рубинами – переменная цикла i - от *0* до *maxRubies*

Поскольку переменные циклов обозначают число мешков, то мы легко определим, сколько в них было драгоценностей. Для этого число сумок с изумрудами нужно умножить на 9, а число сумок с рубинами – на 4. Если в сумме они дадут 59 (*TOTAL*), то задача решена, и можно печатать ответ:

```
   // решаем задачу:
   for var j := 0 to maxEmeralds do
      for var i := 0 to maxRubies do
         if (j * 9 + i * 4 = TOTAL) then
         begin
            Console.WriteLine('Изумрудов: ' + (j * 9));
            Console.WriteLine('Рубинов :  ' + (i * 4));
         end;
```

```
   Console.WriteLine();
end;


begin
   // заголовок окна:
   Console.Title := 'Мешки с сокровищами';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Мешки с сокровищами');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

А ответ такой: рубинов у торговца было 32 штуки.



Повторяющиеся цифры

Росс Хонсбергер в книге **Математические изюминки [XP92]** на страницах 126-127 предлагает решить такую задачу:

> **Определите количество цифр** в длиннейшей последовательности ненулевых одинаковых цифр, которой может оканчиваться полный квадрат, **и найдите наименьший квадрат**, который оканчивается на такую максимальную последовательность.

Элементарная проверка показывает, что квадраты натуральных чисел могут заканчиваться только цифрами **0**, **1**, **4**, **5**, **6** и **9**. Из этого набора мы сразу должны исключить **нуль**.

Дальнейшие рассуждения смотрите в книге. Из них следует, что квадрат не может заканчиваться двумя единицами, пятёрками, шестёрками и девятками. Не может он заканчиваться и четырьмя четвёрками. Следовательно, максимальная последовательность равна **...444**.

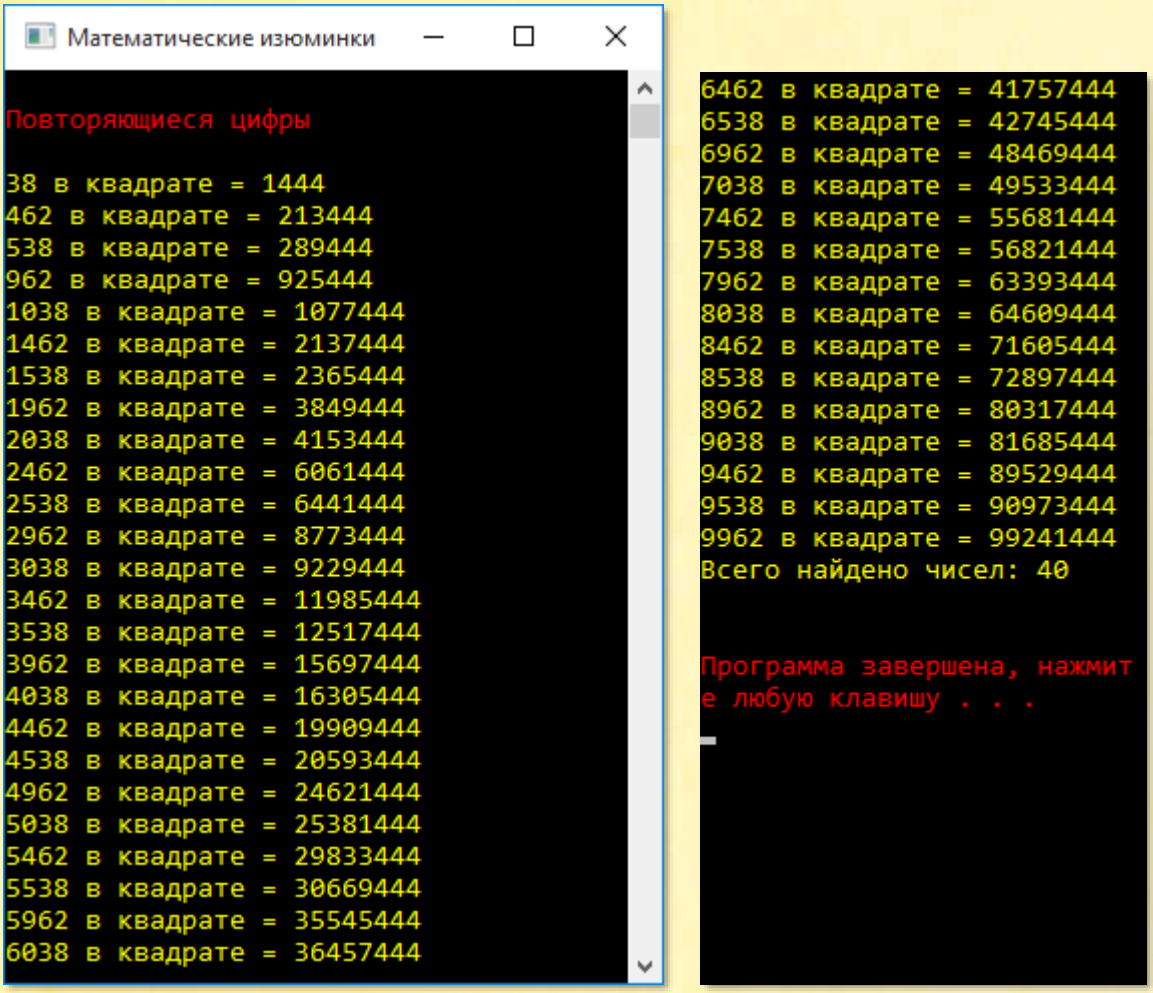
Мы не можем и не должны сомневаться в строгом доказательстве Росса Хонсбергера, поэтому просто напишем программу, которая умеет находить квадраты, оканчивающиеся тремя одинаковыми цифрами (мы не ограничиваем себя только четвёрками).

В главном блоке мы без труда выделяем из квадрата одну и три его последние цифры. Если все они одинаковы, то частное от деления трёх последних цифр на самую последнюю должно равняться 111. Дополнительно мы проверяем, чтобы последняя цифра не оказалась нулём:

```
uses
   System;


// Математические изюминки

// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
```

```
   Console.ForegroundColor := ConsoleColor.Yellow;

   var nVar := 0;
   for i: int64 := 10 to 10000 - 1 do //1000000
   begin
      // последние 3 цифры:
      var n3: int64 := i * i mod 1000;
      // последняя цифра:
      var last: int64 := i * i mod 10;
      if ((last <> 0) and (n3 div last = 111)) then
      begin
         nVar += 1;
         Console.WriteLine('{0} в квадрате = {1}', i, i * i);
      end
   end;
   Console.WriteLine('Всего найдено чисел: ' + nVar);
   Console.WriteLine();
end;


begin
   // заголовок окна:
   Console.Title := 'Математические изюминки';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Повторяющиеся цифры');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

Из миллиона первых квадратов только 4000 заканчиваются на 3 четвёрки. Вот небольшая часть списка:



Квадраты 1,4,9

В книге Айлена Варди (Ilan Vardi) *Computational Recreations in Mathematica* предлагается такое упражнение (*Exercise 2.2*, страница 20) для самостоятельного решения:

> Найдите полные квадраты натуральных чисел, которые состоят только из цифр 1, 4 и 9.

В книге, на страницах 234-237 вы найдёте довольно сложное решение этой задачи для чисел до 10^{42} включительно. Мы упростим себе задачу, ограничив поиски только числами типа *int64*. Впрочем, мы не найдём только 3 числа:

```
9914419419914449449
444411911999914911441
41999499914914994414914994419149444 1
```

```
uses
   System;

type
   int = integer;
   bool = boolean;

// Квадраты 1-4-9

begin
   // заголовок окна:
   Console.Title := 'Computational Recreations in Mathematica';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Квадраты 1,4,9');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

В процедуре Solve мы перебираем все числа от 1 до максимально возможного. Квадрат каждого числа мы проверяем в методе *Test* на предмет наличия в нём посторонних примесей – цифр, отличных от 1, 4 и 9:

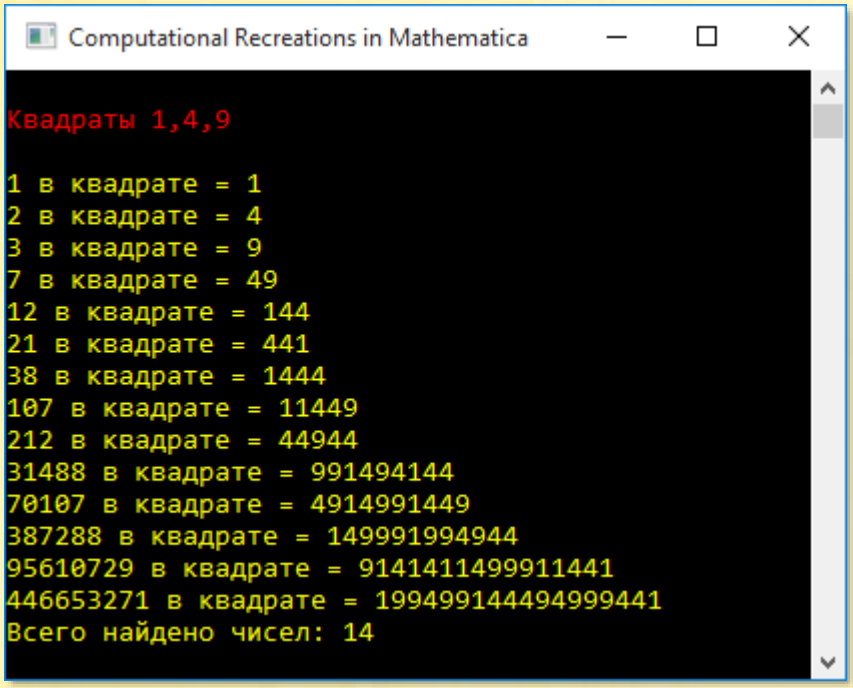
```
// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   var nVar := 0;
   var lmax :=
```

```
   int64.Parse(Math.Truncate(Math.Sqrt(Int64.MaxValue)).ToString());
   for var i: int64 := 1 to lmax do
   begin
      if (Test(i * i)) then
      begin
         nVar += 1;
         Console.WriteLine('{0} в квадрате = {1}', i, i * i);
      end
   end;
   Console.WriteLine('Всего найдено чисел: ' + nVar);
   Console.WriteLine();
end;
```

Проверка очень простая. Мы «отщипываем» от числа его цифры, начиная с хвоста. Как только нам в руки попадётся негодная цифра, мы прекращаем поиски с отрицательным результатом. И только если число целиком и полностью состоит дозволенных цифр, метод возвращает *true*:

```
// ПРОВЕРЯЕМ ЦИФРЫ КВАДРАТА
function Test(num: int64): bool;
begin
   var res := true;
   while (num > 0) do
   begin
      var dig: int64 := num mod 10;
      if ((dig <> 1) and (dig <> 4) and (dig <> 9)) then
      begin
         res := false;
         break;
      end;
      num := num div 10;
   end;
   Result := res;
end;
```

Поиски чисел заняли некоторое время (весьма небольшое для такого рода задач), которое всё же не было потрачено понапрасну – мы нашли 14 (тоже хорошее число, но не квадрат!) квадратов, состоящих из цифр 1, 4 и 9:

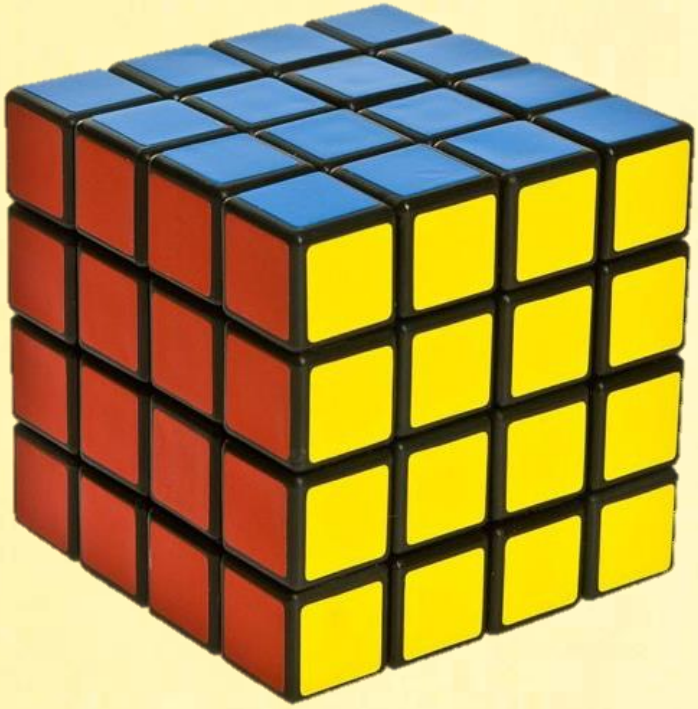


Кубики из кубиков

В журнале *Parade Magazine* от 15 декабря 2013 года предлагается решить такую задачу на пространственное воображение:

> *Consider a cubical structure composed of unit cubes (like a child's building blocks), four on an edge. Without drawing the structure or actually assembling it, how many cubical shapes can you envision within it?*

Иначе говоря, нам предстоит мысленно **пересчитать все возможные кубы** в большом кубе, состоящем из 4 x 4 x 4 маленьких кубиков:



Возьмём самый маленький кубик **1 x 1 x 1**, который находится в левом верхнем углу большого куба. Мы можем передвигать его по осям *X*, *Y* и *Z* от позиции 0 до позиции 4.

Куб **2 x 2 x 2** мы можем перемещать аналогично, но уже только до позиции 3, иначе он выйдет за границы большого куба.

Кубы **3 x 3 x 3** и **4 x 4 x 4** допускают точно такие же действия - с соответствующими ограничениями.

Пусть большой куб имеет размеры сторон в элементарных кубиках **size**, тогда в функции *Solve* мы подсчитываем число кубов заданного размера *1..size* и суммируем в переменной **answer**:

```
uses
   System;

type
```

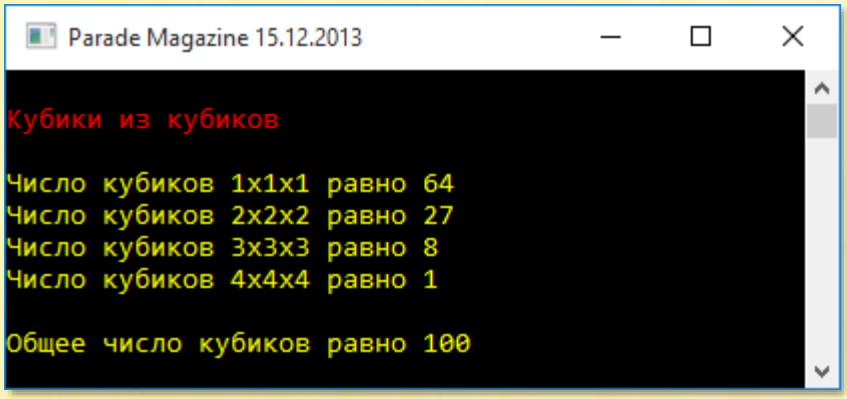
```
   int = integer;
   bool = boolean;

// Кубики из кубиков

begin
   // заголовок окна:
   Console.Title := 'Parade Magazine 15.12.2013';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Кубики из кубиков');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   var size := 4;
   var answer := 0;
   // решаем задачу:
   for var n := 1 to size do
   begin
      var ncube := Solve(size, n);
      answer += ncube;
      Console.WriteLine('Число кубиков {0}x{0}x{0} равно {1}',
                        n, ncube);
   end;
   Console.WriteLine();
   Console.WriteLine('Общее число кубиков равно {0}', answer);

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

Подсчёт кубов заданного размера sizeSmall в кубе sizeBig мы проводим в трёх вложенных циклах *for*:

```
// РЕШАЕМ ЗАДАЧУ
function Solve(sizeBig, sizeSmall: int): int;
begin
   Console.ForegroundColor := ConsoleColor.Yellow;
   var res := 0;
```

```
    for var x := 0 to sizeBig - sizeSmall do
       for var y := 0 to sizeBig - sizeSmall do
          for var z := 0 to sizeBig - sizeSmall do
          begin
             res += 1;
          end;
    Result := res;
end;
```

Всего в кубе 4 x 4 x 4 мы насчитали ровно 100 кубов разных размеров:



Глядя на этот ряд чисел, легко догадаться, что в кубе 5 x 5 x 5 на 125 кубов больше. И так далее...

Бизнес на мышах

В журнале *New Scientist* #1224 от 23 октября 1980 года была напечатана задача *Enigma 81*. *Blind mice*:

Johnny and Willie have hit on a simple scheme to get rich quick. Each has agreed to invest his pocket money in the purchase of a white mouse. Then they will combine their investments and sit back and wait for the multiplier to take effect. A spot of asset stripping will then set them up to plunge into guinea pigs at even greater profit.

The venture is somewhat riskier than they realise. Johnny will be buying his mouse at the Ace Pet Shop, where there is at present a stock of eight white mice, six of which are males and two females. Willie will be using the Peerless Pet Store, where the stock of white mice also numbers eight. Economics being more in their line than biology, each will be buying a mouse at random. But I dare say that all will be well, since it is 11 to 5 on that the mice will be of different sex.

How many females are in stock at the Peerless Pet Store?

На русский язык смысл её можно перевести так:

Джонни и Вилли решили быстро разбогатеть и для этого купили по одной белой мышке (а это оказались морские свинки). Джонни купил мышь в одном магазине, а Вилли в другом. В каждом магазине было по 8 белых мышей. В первом магазине было 6 самцов и 2 самки.

***Сколько было самок во втором магазине**, если Джонни и Вилли выбирали мышей наугад, а вероятность того, что купленные мыши имеют разный пол, равна 11 : 5?*

Вероятность того, что **Джонни** купит **самца**, равна:

$M1 = 6/8$,

а **самку**:

$F1 = 2/8$

Если мы обозначим через **f** число мышек-самок во втором магазине, то вероятность того, что **Вилли** купит самку, равна:

$F2 = f/8$

а **самца**:

$M2 = (8-f)/8$

Вероятность покупки двух самцов равняется произведению вероятностей:

$M1 \times M2$

А вероятность покупки двух самок – произведению вероятностей:

$F1 \times F2$

Общая вероятность покупки однополых мышей равна сумме этих вероятностей:

$p1 = M1 \times M2 + F1 \times F2$

А разнополых:

$p2 = 1 - p1$

Из условия задачи следует, что

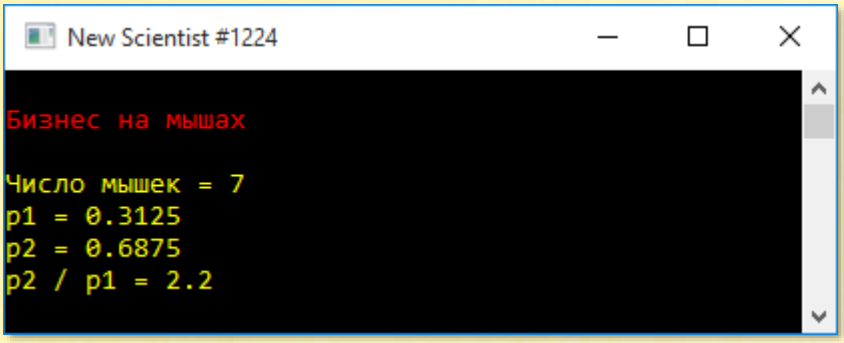
$p2 / p1 = 11 / 5$, то есть

$5*p2 = 11*p1 \rightarrow 5*(1-p1) = 11*p1$

Поскольку число самок во втором магазине может быть от 0 до 8, то легко проверить в цикле *for* все значения и выбрать нужное:

```
uses
   System;

// Бизнес на мышах

// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   for var f := 0 to 8 do
   begin
      var M1 := 6 / 8.0;
      var F1 := 2 / 8.0;
      var M2 := (8 - f) / 8.0;
      var F2 := f / 8.0;
      var p1 := M1 * M2 + F1 * F2;

      if (5 * (1 - p1) = 11 * p1) then
      begin
         Console.WriteLine('Число мышек = ' + f);
         Console.WriteLine('p1 = ' + p1);
         Console.WriteLine('p2 = ' + (1 - p1));
         Console.WriteLine('p2 / p1 = ' + (1 - p1) / p1);
      end
   end;
   Console.WriteLine();
end;


begin
   // заголовок окна:
   Console.Title := 'New Scientist #1224';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Бизнес на мышах');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();
```

```
    Console.WriteLine();
    Console.ForegroundColor := ConsoleColor.Red;
end.
```



Таким образом, во втором магазине было 7 самок и 1 самец.

Безусловно, эту задачу можно решить и без помощи компьютера, но она очень поучительна, если иметь в виду вещественные типы данных: стОит только забыть о десятичной точке в числовом литерале 8.0, как задача не будет решена!

Enigma 1436: One more step

В журнале *New Scientist #2597* от 31-го марта 2007 года была напечатана задача Enigma 1436:

I invite you to consider the following series of numbers: 1, 3, 7, 13, 21 ...

Then find the following:

(a) The six-hundredth member of the series.
(b) A member of the series above the first with less than five digits which is a perfect cube.
(c) A member which is a five-digit palindrome which can also be read as a binary number.
(d) The smaller of the two consecutive members which are 1000 apart.

Рассмотрим следующий ряд чисел: 1, 3, 7, 13, 21 …

Найдите:

(a) Шестисотый член ряда.
(b) Член ряда, который:
больше первого,
состоит менее чем из пяти цифр и
является кубом целого числа.
(c) Член ряда, который:
является палиндромом с пятью цифрами,
может быть прочитан как двоичное число.
(d) Два наименьших последовательных члена ряда, разность между которыми равняется 1000.

Прежде чем решать задачу, мы должны научиться вычислять члены ряда.

Обозначим ряд буквой **a**, тогда первый член равняется:

$$a_1 = 1$$

Обозначим разность между членами ряда буквой **d**, тогда:

$$d_1 = 3-1 = 2$$
$$d_2 = 7-3 = 4$$
$$d_3 = 13-7 = 6$$
$$d_4 = 21-13 = 8$$
$$\cdot\ \cdot\ \cdot$$
$$d_n = 2 * n,$$

где **n** – номер члена ряда.

Теперь мы легко найдём любой член ряда:

$$a_n = a_{n-1} + d_n$$

Для вывода **нерекуррентной** формулы воспользуемся *методом исчисления конечных разностей*, который описан в книге Мартина Гарднера **[ГМ72]**, на страницах 81-95, а также в книге *Как решать комбинаторные задачи на компьютере*:

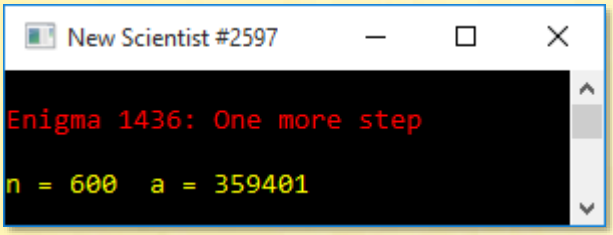
```
1  3  7  13  21
 2  4  6   8
  2  2  2

Откуда:

a_n = n^2 - n + 1
(1)
```

Переходим к решению задачи и в главном блоке программе вызываем функцию *Solve*, которая и даст нам ответы на все вопросы:

```
uses
   System;

type
   int = integer;
   bool = boolean;

// Enigma 1436. One more step

begin
   // заголовок окна:
   Console.Title := 'New Scientist #2597';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Enigma 1436: One more step');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

Подзадачу (a) мы решаем, просто подставив номер члена 600 в формулу (1):

```
// ВЫЧИСЛЯЕМ n-ный ЧЛЕН РЯДА
function GetAn(n: int): int;
begin
  Result := n * n - n + 1;
end;

// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
    Console.ForegroundColor := ConsoleColor.Yellow;

    // (a)
    Console.WriteLine('n = {0,3}  a = {1,3}', 600, GetAn(600));
    Console.WriteLine();
end;
```



Для решения подзадачи (b) нам потребуется функция IsQube, которая умеет определять, является ли заданное целое число точным кубом. Все условия подзадачи (b) легко записать внутри бесконечного цикла *while*:

```
    // (b)
    var n := 1;
    while (true) do
    begin
       var a := GetAn(n);
       if ((n > 1) and (a <= 9999) and (IsQube(a))) then
       begin
          Console.WriteLine('n = {0,3} a = {1,4} cube = {2,2} ', n,
                            a, Math.Pow(a, 1.0 / 3));
          break;
       end;
       n += 1;
    end;
```

Здесь:

- Номер **n** искомого члена ряда больше единицы:

```
n > 1
```

- Искомый член **a** ряда состоит менее чем из пяти цифр:

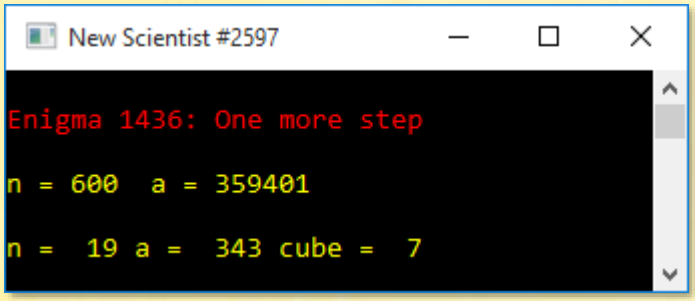
```
a <= 9999
```

- Искомый член ряда **a** – точный куб:

```
IsQube(a)
```

```
// ОПРЕДЕЛЯЕМ ЯВЛЯЕТСЯ ЛИ ЗАДАННОЕ ЧИСЛО
// ПОЛНЫМ КУБОМ
function IsQube(num: int64): bool;
begin
   var r := Trunc(Math.Round(Math.Pow(num, 1.0 / 3.0)));
   Result := r * r * r = num;
end;
```

Поскольку мы заранее не знаем, какой член ряда удовлетворяет этим условиям, то запускаем бесконечный цикл, который прерываем сразу же, как только найдём заданный член ряда:



Подзадачу **(c)** мы решаем аналогично, но условие прекращения цикла должно включать проверки, что очередной член ряда:

- пятизначное число
- палиндром
- состоит только из цифр 0 и 1:

```
   Console.WriteLine();
   // (c)
   n := 0;
   while (true) do
   begin
      n += 1;
      var a := GetAn(n);
      if (a > 99999) then break;
      if (a < 10000) then continue;
      var s := a.ToString();
      var r := new string(s.ToCharArray().Reverse().ToArray());
      if ((s.IndexOfAny(new char[]('2', '3', '4', '5', '6', '7',
                                   '8', '9' )) = -1) and (s = r))
then
      begin
         Console.WriteLine('n = {0,4} a = {1,4} ', n, s);
         break;
      end
   end;
```

Здесь:

- Искомый член **a** ряда состоит из пяти цифр:

```
if (a > 99999)
if (a < 10000)
```

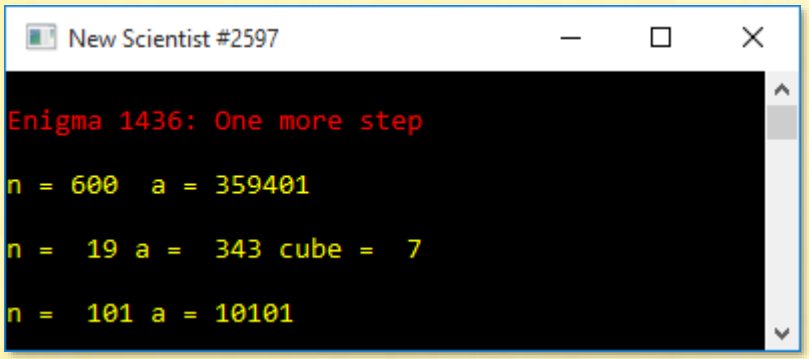
- Искомый член ряда **a** – палиндром:
- Искомый член ряда состоит из цифр 0 и 1:

```
var s := a.ToString();
var r := new string(s.ToCharArray().Reverse().ToArray());
if ((s.IndexOfAny(new char[]('2', '3', '4', '5', '6', '7',
                             '8', '9' )) = -1) and (s = r))
```

С помощью метода IndexOfAny мы легко определим, имеются ли в заданном числе (точнее – в строке, полученной из этого числа) «запрещённые» цифры.

> На самом деле многие проверки в данном случае излишни; тот же самый ответ мы получили бы и с гораздо меньшими усилиями.

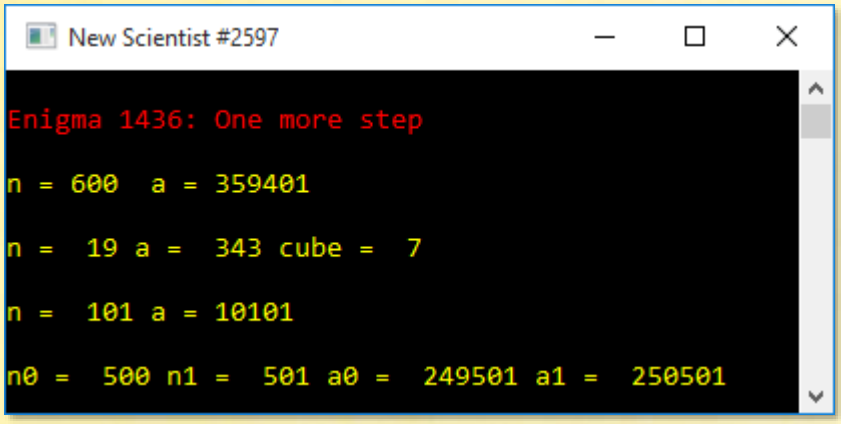
Все условия соблюдены, запускаем программу – и находим решение подзадачи (c):



И последнюю подзадачу мы решаем без труда:

```
Console.WriteLine();
// (d)
n := 1;
while (true) do
begin
   var a0 := GetAn(n);
   var a1 := GetAn(n + 1);
   if (a1 - a0 = 1000) then
   begin
      Console.WriteLine('n0 = {0,4} n1 = {1,4} a0 = {2,7}
                         a1 = {3,7}',  n, n + 1, a0, a1);
      break;
   end;
   n += 1;
end;
```

Если мы начнём с первого члена ряда (*n=1*), то нам нужно найти разность между ним и *следующим* членом ряда, номер которого на единицу больше. Если эта разность не равняется 1000, то переходим ко второму члену ряда и находим разность между ним и третьим членом ряда. Продолжаем так до тех пор, пока не получим заданную разность:



Три мушкетёра

Эту задачу я придумал много лет тому назад под влиянием подросткового романа Александра Дюма о друзьях-мушкетёрах:

Каждый, кто читал о похождениях знаменитых мушкетеров Атоса, Портоса и Арамиса, знает, что все они любили подраться на дуэли и слегка приврать, рассказывая о своих победах. А ещё они были не дураки сыграть в дурака. И вот однажды приключилась с ними на этом поприще удивительная история, которую поведали они шевалье д'Артаньяну вечером того самого дня, когда они чуть было не наkололи друг друга на шпаги.

- Так вот, любезный д'Артаньян, - начал Атос, - в прошлом году в это же самое время довелось нам охранять королевский дворец - без малого целую неделю. Кажется, эти бунтовщики из Ла-Рошели послали убийцу к нашему дорогому Людовику.

И по 8 часов в сутки мы стояли в карауле, а всё остальное время бездельничали в казарме…

- Ты ещё молод, д'Артаньян, - добавил Арамис, и не можешь даже представить себе, какое это было унылое время! За воротами Лувра вовсю кипела и била ключом жизнь, рекой лилось бургундское, а мы буквально не знали, чем себя занять…

- Вот именно! – прогремел Портос и с размаху ударил кружкой по столу. – Почти два дня мы изнывали от тоски, пока наш учёный Арамис не предложил сыграть в карты. Он научил нас играть в подкидного дурака. Отличная игра! Особенно, когда после игры мы подкидывали вверх очередного дурака и из его карманов сыпались деньги! Ну, это, правда, было позже. А для начала мы сыграли между собой десяток-другой партий и так наловчились бить и крыть, что решили уже играть на деньги. Своих-то денег у нас тогда не было, поэтому мы и удумали играть на чужие. И за три дня мы обыграли весь наш полк!

Арамис в первый день выиграл 23 пистоля, во второй – 38, а в третий – 41. Атос играл ещё удачнее и выиграл 44, 50 и 85 пистолей. А мне так вообще везло просто неприлично! Уже в первый день я выиграл больше всех – 89 пистолей, во второй – 163, а в третий – целых 206 пистолей!

- Постой! – возмущённо крикнул Арамис. – Ты что-то слишком много себе насчитал! 206-то пистолей выиграл я! И было это не в третий день, а…

- Да нет, мой дорогой Арамис! – раздражённо возразил ему Портос. – 206 пистолей выиграл я и было это именно в третий день!

И тут они начали слегка возбуждённо объяснять и доказывать свою правоту, так что вскоре уже готовы были вызвать друг друга на дуэль. Но тут их остановил мудрый Атос:

- Не спорьте, друзья мои, - сказал он, - я не помню, кто выиграл 206 пистолей (тут он незаметно подмигнул д'Артаньяну), но когда мы подсчитали деньги, то оказалось, что я выиграл ровно в два раза больше, чем Арамис. Это меня сильно удивило, поэтому я это точно запомнил.

И, по-моему, этого вполне достаточно для того, чтобы определить, сколько пистолей выиграл каждый из нас.

- Ха! – по-провинциальному просто воскликнул д'Артаньян. – Я тут закончил подсчёты, - и он показал на поля своей новой шляпы, - и вот какая у меня получилась картина. На самом-то деле 206 пистолей выиграл…

Всем известно, что д'Артаньян никогда не учился в школе, поэтому мы вряд ли можем доверять его шляпным расчётам. Зато всякий, кто в школе учился, без труда сможет решить, кто – Портос или Арамис – был прав, утверждая, что он выиграл за один день 206 пистолей, и вообще - кто сколько денег выиграл за три дня. Д'Артаньяну, кажется, это удалось определить. Может быть, удастся и вам. А если нет – значит, всё дело было в шляпе.

Нам предстоит поделить на 3 кучки 9 денежных сумм в пистолях:

23, 38, 41, 44, 50, 85, 89, 163, 206

При этом эти суммы не повторяются, Атос получает вдвое больше Арамиса, ну а Портосу достанется то, что останется.

Пистоли для удобства пользования мы поместим не в длинный чулок и даже не в кошелёк, а в множество pistols типа *HashSet*:

```
uses
   System;

type
   int = integer;
   bool = boolean;

// Три мушкетёра

begin
   // заголовок окна:
   Console.Title := 'Головоломное ассорти';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Три мушкетёра');
   Console.ForegroundColor := ConsoleColor.Green;
```

```
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.

// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   var pistols := new HashSet<int>(Seq(23, 38, 41, 44, 50,
                                       85, 89, 163, 206 ));
```

Так как порядок выигрышей нас не интересует, то для начала мы выберем 3 суммы для Арамиса, и сделаем это так, что каждая следующая будет больше предыдущей:

```
   // Арамис:
   for var aramis1 := 0 to pistols.Count - 1 do
      for var aramis2 := aramis1 + 1 to pistols.Count - 1 do
         for var aramis3 := aramis2 + 1 to pistols.Count - 1 do
         begin
```

Так мы избежим проверок на совпадение сумм и многочисленных повторов при разбиении множества на подмножества.

Из оставшихся 6 сумм мы аналогично выбираем 3 суммы для Атоса. Но теперь мы должны учитывать, что Атос не мог выиграть те же самые суммы, что и Арамис:

```
// Атос:
for var atos1 := 0 to pistols.Count - 1 do
begin
   if ((atos1 = aramis1) or (atos1 = aramis2) or
       (atos1 = aramis3)) then
      continue;
   for var atos2 := atos1 + 1 to pistols.Count - 1 do
   begin
```

```
      if ((atos2 = aramis1) or (atos2 = aramis2) or
          (atos2 = aramis3)) then
         continue;
      for var atos3 := atos2 + 1 to pistols.Count - 1 do
      begin
         if ((atos3 = aramis1) or (atos3 = aramis2) or
             (atos3 = aramis3)) then
            continue;
```

Определив возможные выигрыши для Арамиса и Атоса, мы можем проверить условие задачи, что Атос выиграл вдвое больше Арамиса:

```
if (pistols.ElementAt(atos1) + pistols.ElementAt(atos2) +
                               pistols.ElementAt(atos3) <>
                               2 * (pistols.ElementAt(aramis1) +
                               pistols.ElementAt(aramis2) +
                               pistols.ElementAt(aramis3))) then
    continue;
```

Если это не так, мы продолжаем делить выигрыши иначе.

Если же все условия соблюдены, мы печатаем суммы, выигранные Арамисом и Атосом:

```
Console.WriteLine('Арамис: ' + pistols.ElementAt(aramis1) + ' ' +
                               pistols.ElementAt(aramis2) + ' ' +
                               pistols.ElementAt(aramis3));
Console.WriteLine('Атос:   ' + pistols.ElementAt(atos1) + ' ' +
                               pistols.ElementAt(atos2) + ' ' +
                               pistols.ElementAt(atos3));
```

Чтобы найти выигрыши Портоса, мы помещаем выигрыши Арамиса и Атоса в массив ps:

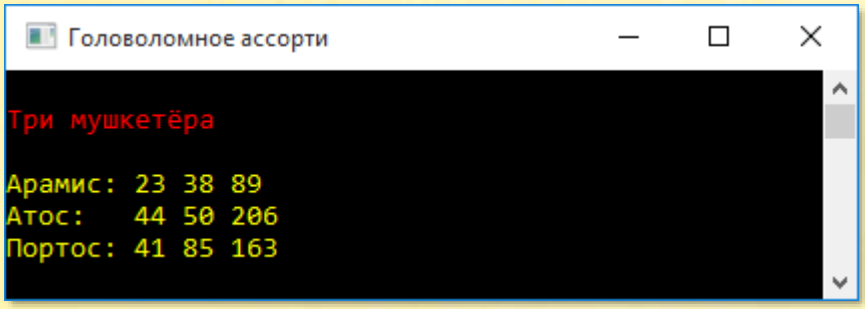
```
   // Портос:
   var ps := new int[](pistols.ElementAt(aramis1),
                       pistols.ElementAt(aramis2),
                       pistols.ElementAt(aramis3),
```

```
                        pistols.ElementAt(atos1),
                        pistols.ElementAt(atos2),
                        pistols.ElementAt(atos3));
```

А затем находим денежные суммы, которые останутся в множестве pistols, из которого мы исключили выигрыши Арамиса и Атоса:

```
                    Console.Write('Портос: ');
                    var portos := pistols.Except(ps);
                    foreach var p in portos do
                        Console.Write(p + ' ');

                    Console.WriteLine();
                end
            end
        end
    end
end;
```

Как показывает рисунок, выигрыш Арамиса составил 23 + 38 + 89 = 150 пистолей, Атос сыграл в 2 раза лучше Арамиса и выиграл 44 + 50 + 206 = 300 пистолей, а на долю Портоса остаётся 41 + 85 + 163 = 289 пистолей. Так что 206 пистолей не выигрывал ни Портос, ни Арамис.



Дружба - дружбой, а пистоли - врозь!
Кардинал Ришелье

Vier Gewichte

Задача 121 из книги головоломок *Gehirnjogging*. Вот условие задачи – сначала на немецком языке, а затем на русском:

Задача 121. Vier Gewichte.

Professor Busch hat eine Waage und vier Gewichte. Mit diesen Gewichten kann er jedes Gewicht in vollen Kilogramm bis 40 kg bestimmen.

Welche vier Gewichte stehen Professor Busch zur Verfügung?

Четыре гири

У профессора Буша (не путать с американским президентом!) имеются рычажные весы с двумя чашами и четыре гири, с помощью которых он может определить вес от 1 до 40 кг в целых числах.

Какие четыре гири имеет профессор Буш?

Для удобства обозначим веса гирь буквами a, b, c и d.

Сначала мы рассмотрим комбинации гирь, которые можно положить на одну чашу весов. В этом случае на вторую чашу следует положить груз такого же веса, чтобы взвесить его.

Итак, на одну чашу весов можно положить любую из четырёх гирь:

1. a
2. b
3. c
4. d

Но можно положить и любую пару гирь:

5. a + b
6. a + c
7. a + d
8. b + c
9. b + d
10. c + d

Или любую **тройку** гирь:

11. b + c + d
12. a + c + d
13. a + b + d
14. a + b + c

Или, наконец, все **четыре** гири:

15. a + b + c + d

В лучшем случае мы получим 15 разных весов, что гораздо меньше требуемых 40. Поэтому будем раскладывать гири всеми возможными способами на две чаши. Тогда взвешиваемый груз нужно положить на ту чашу весов, на которой общий вес гирь меньше.

Начнём с **двух** гирь. Одну из них мы кладём на левую чашу весов, вторую на правую. Вес груза равен абсолютной величине разности весов гирь:

16. Abs(a - b)
17. Abs(a - c)
18. Abs(a - d)
19. Abs(b - c)
20. Abs(b - d)
21. Abs(c - d)

Три гири между двумя чашами можно поделить так: на левую кладём 2 гири, на правую - одну. Вес груза равен абсолютному значанию разности весов гирь на левой и правой чашках:

22. Abs(a + b - c)
23. Abs(a + b - d)
24. Abs(a + c - b)
25. Abs(a + c - d)
26. Abs(a + d - b)
27. Abs(a + d - c)
28. Abs(b + c - a)
29. Abs(b + c - d)
30. Abs(b + d - a)
31. Abs(b + d - c)
32. Abs(c + d - a)
33. Abs(c + d - b)

Четыре гири можно сгруппировать двумя способами.

Первый способ: на левую чашу кладём 1 гирю, на правую – 3:

34. Abs(a - b - c - d)
35. Abs(b - a - c - d)
36. Abs(c - a - b - d)
37. Abs(d - a - b - c)

Второй способ: на левую и правую чаши кладём по 2 гири:

38. Abs(a + b - c - d)
39. Abs(a + c - b - d)
40. Abs(a + d - b - c)
41. Abs(b + c - a - d)
42. Abs(b + d - a - c)
43. Abs(c + d - a - b)

Таким образом, как бы мы ни перекладывали гири, мы не получим более 43 разных весов. Этого вполне достаточно для взвешивания грузов от 1 до 40 кг, если нам удастся правильно подобрать вес гирь.

Так как только 3 комбинации весов могут повторяться, то:

- ни одна гиря не тяжелее 40 килограммов, иначе число комбинаций весов окажется менее 40
- по этой же причине никакая пара гирь не тяжелее 40 килограммов
- все гири имеют разный вес

Пишем программу:

```
uses
   System;

type
   int = integer;

// Vier Gevichte

begin
   // заголовок окна:
   Console.Title := 'Gehirnjogging, Задача 121';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Vier Gevichte');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

Проведённые нами разыскные мероприятия позволяют нам без труда решить эту переборную задачу с помощью четырёх вложенных циклов *for*:

```
  for var a := 1 to 40 do
      for var b := a + 1 to 40 - a do
         for var c := b + 1 to 40 - a - b do
            for var d := c + 1 to 40 - a - b - c do
            begin
```

Здесь мы полагаемся на грубую силу, так как перебор не столь велик, чтобы по ходу перебора отвергать заведомо негодные комбинации весов гирь.

Так как нам нужно получить 40 разных весов, то мы должны их подсчитывать так, чтобы не было повторов. В таких случаях обычно используют множества, которые могут содержать только неповторяющиеся значения:

```
// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   var hs := new HashSet<int>();
```

В начале внутреннего цикла мы удаляем из множества все элементы, а затем наполняем его новыми значениями, формулы для расчёта которых мы вывели ранее:

```
 hs.Clear();

 hs.Add(a);
 hs.Add(b);
 hs.Add(c);
 hs.Add(d);

 hs.Add(a + b);
 hs.Add(a + c);
 hs.Add(a + d);
 hs.Add(b + c);
 hs.Add(b + d);
 hs.Add(c + d);

 hs.Add(b + c + d);
 hs.Add(a + c + d);
 hs.Add(a + b + d);
 hs.Add(a + b + c);

 hs.Add(a + b + c + d);
```

```
hs.Add(Abs(a - b));
hs.Add(Abs(a - c));
hs.Add(Abs(a - d));
hs.Add(Abs(b - c));
hs.Add(Abs(b - d));
hs.Add(Abs(c - d));

hs.Add(Abs(a + b - c));
hs.Add(Abs(a + b - d));
hs.Add(Abs(a + c - b));
hs.Add(Abs(a + c - d));
hs.Add(Abs(a + d - b));
hs.Add(Abs(a + d - c));
hs.Add(Abs(b + c - a));
hs.Add(Abs(b + c - d));
hs.Add(Abs(b + d - a));
hs.Add(Abs(b + d - c));
hs.Add(Abs(c + d - a));
hs.Add(Abs(c + d - b));

hs.Add(Abs(a - b - c - d));
hs.Add(Abs(b - a - c - d));
hs.Add(Abs(c - a - b - d));
hs.Add(Abs(d - a - b - c));

hs.Add(Abs(a + b - c - d));
hs.Add(Abs(a + c - b - d));
hs.Add(Abs(a + d - b - c));
hs.Add(Abs(b + c - a - d));
hs.Add(Abs(b + d - a - c));
hs.Add(Abs(c + d - a - b));
```

Чтобы нас не терзали смутные сомнения, мы отбираем в коллекцию *gewichte* только веса, заключённые между 1 и 40:

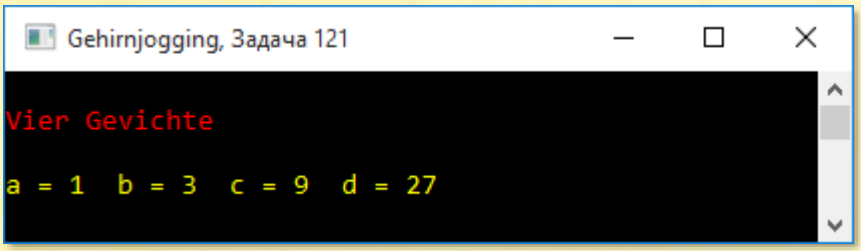
```
var gewichte := hs.Where(g -> (g >= 1) and (g <= 40));
```

Можете проверить, что эта операция лишняя, но это не очевидно.

Когда в наборе окажется не менее 40 элементов (а больше их и не будет!), мы печатаем ответ на экране:

```
                if (gewichte.Count >= 40) then
                 begin
                    Console.WriteLine('a = {0}  b = {1}  c = {2}
                                       d = {3}', a, b, c, d);
                 end;
              end;
   Console.WriteLine();
end;
```

По данным на рисунке легко догадаться, что вес гирь выражается степенями тройки: 3^0, 3^1, 3^2, 3^3, то есть для решения нужно использвать троичную систему счисления.



Тот, кто уже решал задачи на взвешивание, твёрдо это знает. А мы, увы, не знали...

Гимнастический зал

Задача из книги *Увлекательная математика*:

В гимнастическом зале стоит несколько одинаковых по длине скамей. Если спортсмены попытаются сесть по 6 человек на скамью, то одна скамья окажется незаполненной: на ней сядут лишь 3 спортсмена. Если же спортсмены попытаются сесть по 5 человек на скамью, то 4 спортсменам места не хватит.

Сколько спортсменов и сколько скамей в гимнастическом зале?

Пусть *x* – число спортсменов, а *y* – число скамей.

В первом случае *x* спортсменов усядутся по 6 человек на *(y - 1)* скамью, а на одну оставшуюся – ещё трое:

$$6(y - 1) + 3 = x \quad (1)$$

Во втором случае *x* спортсменов усядутся по 5 человек на y скамей, и ещё четверым места не хватит:

$$5y + 4 = x \quad (2)$$

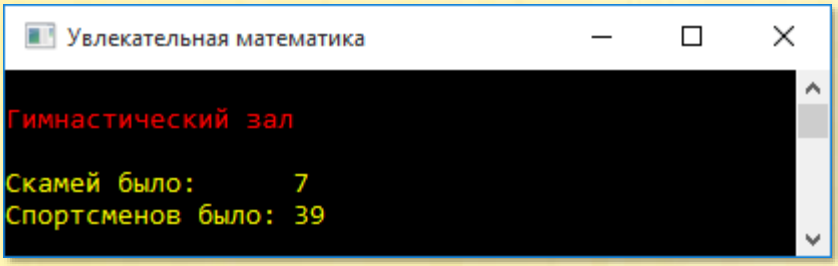
Получили простейшую **систему из двух уравнений**, которая решается перебором целых значений в бесконечном цикле *for*. Это вполне возможно, поскольку и число скамей, и число спортсменов выражаются **целыми** числами:

```
uses
   System;

type
   int = integer;
   bool = boolean;

// Увлекательная математика

// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   for var skamey := 1 to int.MaxValue do
   begin
        var uslovie1 := 6 * (skamey - 1) + 3;
        var uslovie2 := 5 * skamey + 4;

        if (uslovie1 = uslovie2)then
        begin
```

```
            Writeln ('Скамей было:      ' + skamey);
            Writeln ('Спортсменов было: ' + uslovie1);
            break;
        end;
    end;

    Console.WriteLine();
end;


begin
    // заголовок окна:
    Console.Title := 'Увлекательная математика';
    Console.WriteLine('');
    Console.ForegroundColor := ConsoleColor.Red;
    Console.WriteLine('Гимнастический зал');
    Console.ForegroundColor := ConsoleColor.Green;
    Console.WriteLine();

    Solve();

    Console.WriteLine();
    Console.ForegroundColor := ConsoleColor.Red;
end.
```

Под **условиями** мы понимаем здесь правую часть уравнений (1) и (2), то есть число спортсменов.

Ответ на задачу такой:



Грузовые машины

Задача из книги *Увлекательная математика*:

Двумя грузовыми машинами требуется перевезти 143 т сыра. Грузоподъёмность одной машины в 1,5 раза больше, чем другой. Для перевозки всего груза полностью гружёной машине меньшей грузоподъёмности понадобится совершить 31 рейс и 27 рейсов машине большей грузоподъёмности.

Сколько тонн сыра перевозит за 1 рейс каждая машина?

Обозначим буквой *x* грузоподъёмность первой машины, а буквой *y* – второй.

Тогда: $y = 1.5x$

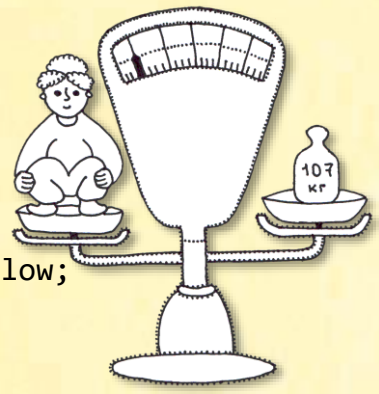
Первая машина сделала 31 рейс и перевезла 31x тонн сыра.
Вторая машина сделала 27 рейсов и перевезла 27y тонн сыра.
Обе машины вместе перевезли 143 тонны сыра:

$31x + 27y = 143$

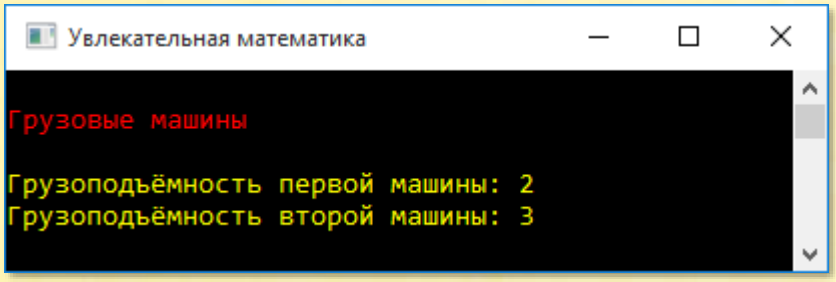
Мы получили элементарную **систему из двух линейных уравнений**, с которой элементарно справился бы и доктор Ватсон:

```
uses
   System;

type
   int = integer;
```

```
   bool = boolean;

// Увлекательная математика

// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   for var mashina1 := 1 to int.MaxValue do
   begin
        var mashina2 := 1.5 * mashina1;

        if (31 * mashina1 + 27 * mashina2 = 143) then
        begin
            Writeln ('Грузоподъёмность первой машины: ' + mashina1);
            Writeln ('Грузоподъёмность второй машины: ' + mashina2);
            break;
        end;
   end;

   Console.WriteLine();
end;

begin
   // заголовок окна:
   Console.Title := 'Увлекательная математика';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Грузовые машины');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```



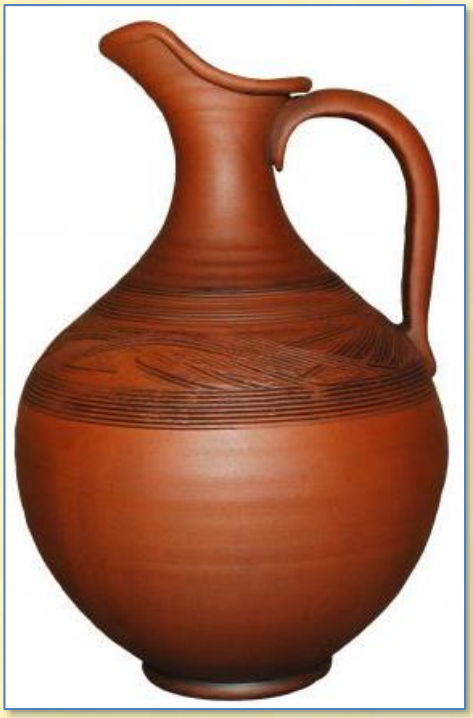
Грузоподъёмность найдена:



Кувшин

Задача 22 из книги *Математический фольклор*:

> *Четыре чашки и один кувшин для воды весят столько, сколько 17 свинцовых шариков. Кувшин весит столько, сколько одна чашка и 7 шариков.*
>
> *Сколько шариков уравновешивают кувшин?*



Вполне естественно обозначить вес кувшина в свинцовых шариках через *x*. Тогда на долю чашки достанется *y* шариков.

По условию задачи:

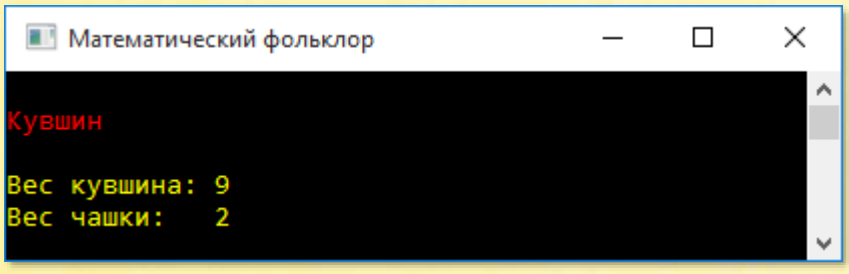
$x = y + 7$
$x + 4y = 17$

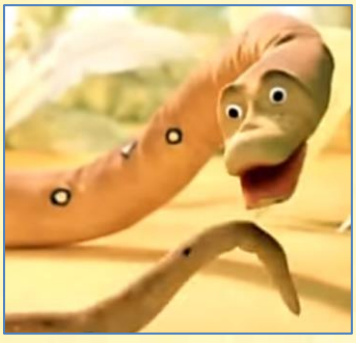
В программе вместо переменной *x* мы используем переменную **kuvshin**, а вместо *y* - **chashka**. В «бесконечном» цикле ***for*** мы изменяем вес чашки, находим вес кувшина и проверяем выполнение условия задачи:

```
uses
   System;

type
   int = integer;
   bool = boolean;

// Математический фольклор

// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;
```

```
    for var chashka := 1 to int.MaxValue do
    begin
        var kuvshin := chashka + 7;

        if (kuvshin + 4 * chashka = 17) then
        begin
            Writeln ('Вес кувшина: ' + kuvshin);
            Writeln ('Вес чашки:   ' + chashka);
            break;
        end;
    end;

    Console.WriteLine();
end;

begin
    // заголовок окна:
    Console.Title := 'Математический фольклор';
    Console.WriteLine('');
    Console.ForegroundColor := ConsoleColor.Red;
    Console.WriteLine('Кувшин');
    Console.ForegroundColor := ConsoleColor.Green;
    Console.WriteLine();

    Solve();

    Console.WriteLine();
    Console.ForegroundColor := ConsoleColor.Red;
end.
```

Если измерять вес не в килограммах, а в свинцовых шариках, то ответ на задачу такой:





Вспоминается фраза из весёлого мультика: «а в попугаях-то я гораздо длиннее!».

Ещё один кувшин

Задача 24 из книги *Математический фольклор*:

> *Пусть 2 чашки и 2 кувшина весят столько, сколько 14 блюдец. Один кувшин весит столько, сколько 1 чашка и 1 блюдце.*
>
> *Сколько блюдец уравновесят 1 кувшин?*

Чтобы нас слегка запутать и смутить, вес кувшина измеряется в этой задачке не в шариках, а в блюдцах.

Пусть кувшин весит *x* блюдец, а чашка - *y* блюдец. Тогда:

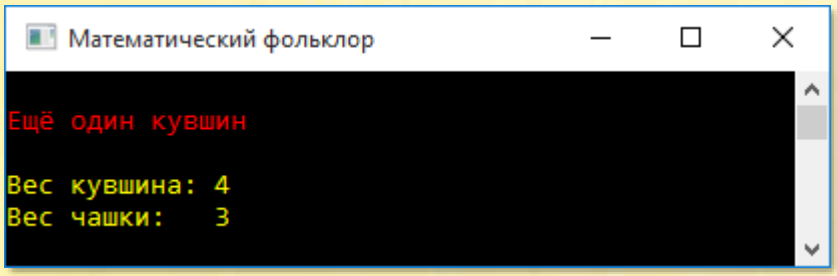
$x = y + 1$
$2y + 2x = 14$

«Перефразируя» предыдущую кувшинную задачу, получаем:

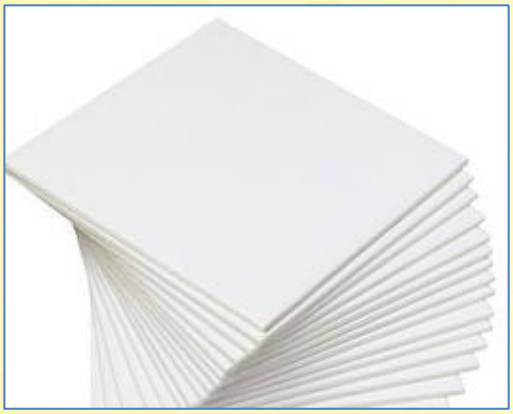
```
uses
   System;

type
   int = integer;
   bool = boolean;

// Математический фольклор

// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   for var chashka := 1 to int.MaxValue do
```

```
    begin
        var kuvshin := chashka + 1;

        if (2 * kuvshin + 2 * chashka = 14) then
        begin
            Writeln ('Вес кувшина: ' + kuvshin);
            Writeln ('Вес чашки:   ' + chashka);
            break;
        end;
    end;

    Console.WriteLine();
end;


begin
    // заголовок окна:
    Console.Title := 'Математический фольклор';
    Console.WriteLine('');
    Console.ForegroundColor := ConsoleColor.Red;
    Console.WriteLine('Ещё один кувшин');
    Console.ForegroundColor := ConsoleColor.Green;
    Console.WriteLine();

    Solve();

    Console.WriteLine();
    Console.ForegroundColor := ConsoleColor.Red;
end.
```

Вес посуды в блюдцах такой:



Складывание бумаги



Как иногда удивительным образом стародавние задачи появляются в новом обличии! Вы, конечно, помните знаменитую шахматную задачу. Казалось бы, вполне современная **головоломка со складыванием листа бумаги** не имеет с ней родственных связей, но это не так. Обе поражающие воображение задачи основаны на свойствах **показательной функции**.

В 28-ом математическом выпуске *Академии занимательных наук* профессор Круглов наглядно доказывает хомячку Циркулю, что лист бумаги формата А4 нельзя сложить вдвое более **6 раз**:

Но в *Интернете* можно найти ролик, в котором показано, как такой же лист бумаги можно сложить **7 раз**:

Фокус в том, что сначала листок нужно складывать по длинной стороне, а не попеременно по длинной – по короткой.

Нашлись умельцы, которые взяли огромный лист и свернули его 9 раз:

Давайте разбираться, почему бумага так упорно не желает складываться вдвое!

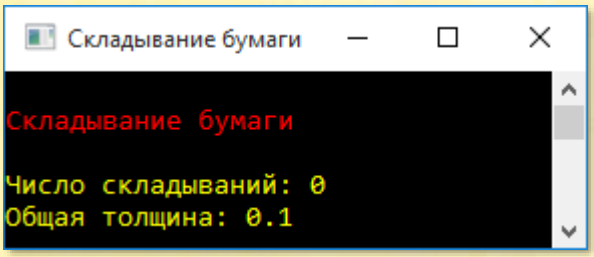
Так как мы не проводим натурные испытания, то довольствуемся простой **математической моделью** процесса складывания бумаги вдоль и поперёк. Она несущественно упрощает этот процесс, зато описывается элементарной формулой:

$d = a \cdot 2^n$

Здесь:

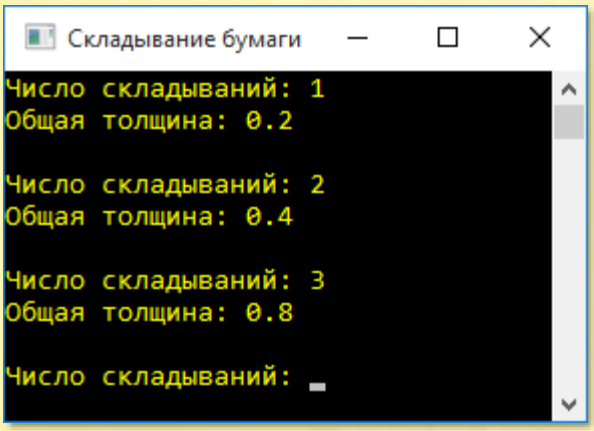
d – общая толщина сложенного листа бумаги
a – толщина самого листа
n – число складываний

Возьмём достаточно тонкую бумагу толщиной **a = 0.1** мм. В начале эксперимента весь «свёрток» имеет такую же толщину:

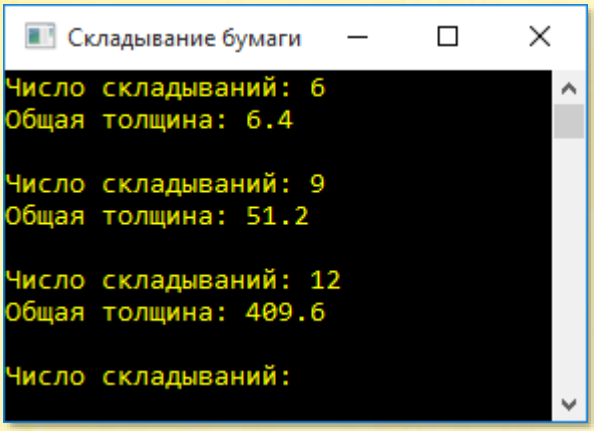


Пока всё нормально.

После трёх складываний толщина сложенного листа бумаги всё ещё меньше 1 миллиметра:



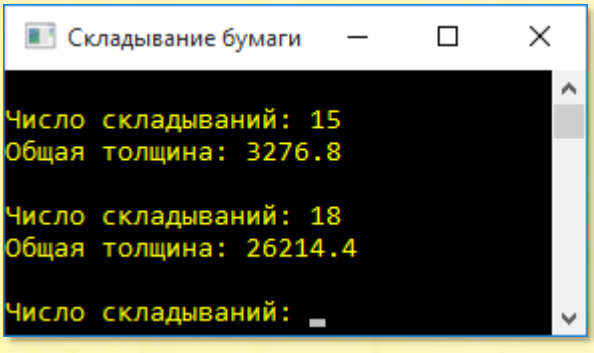
Но при последующих складываниях толщина листа стремительно растёт:



После 9 складываний толщина достигнет 5 сантиметров, а после 12 – почти 41 сантиметра. Теперь легко представить размеры исходного листа бумаги, чтобы его можно было сложить пополам при такой толщине.

Если конечная площадь верхней поверхности листа 40 квадратных сантиметров, то исходный лист при 12 складываниях должен быть в 4096 раз больше, то есть 163840 квадратных сантиметров, или 16,4 квадратных метра. На самом деле площадь исходного листа должна быть значительно больше, поскольку мы складываем не отдельные листы стопкой, а единственный лист, часть которого образует боковые поверхности. Также мы предполагаем, что лист размером 40 x 80 сантиметров толщиной 40 сантиметров ещё можно сложить вдвое.

Но если 12 раз сложить вдвое огромный лист тонкой (!) бумаги ещё вполне возможно, то дальше толщина листа будет измеряться метрами:



Здесь мы наблюдаем ту же самую картину, что и при выкладывании зёрен на шахматную доску. Толщина листа бумаги растёт так быстро, что даже 15 складываний проделать не удастся.

В книге Вальтера Литцмана В*еликаны и карлики в мире чисел*, пятое издание которой вышло в Лейпциге в 1953 году, эксперимент со складыванием бумаги приводится как пример быстрого роста показательной функции. При складывании листа бумаги толщиной 1/10 мм 40 раз высота «стопки» превысит 100 000 километров!

А вот совсем короткая программа, которая помогла нам провести «мысленный» эксперимент:

```
uses
   System;

type
   int = integer;
   bool = boolean;

// Складывание бумаги

// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   var tolshinaLista := 0.1;
   var nSklad := 0;
   while(true) do
   begin
        Write ('Число складываний: ');
        Read(nSklad);
        if (nSklad < 0) then
            break;
        var tolshina := tolshinaLista * power(2, nSklad);
        Writeln ('Общая толщина: ' + tolshina);
        Writeln ();
   end;

   Console.WriteLine();
end;
```

```
begin
   // заголовок окна:
   Console.Title := 'Складывание бумаги';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Складывание бумаги');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

Bogenschießen

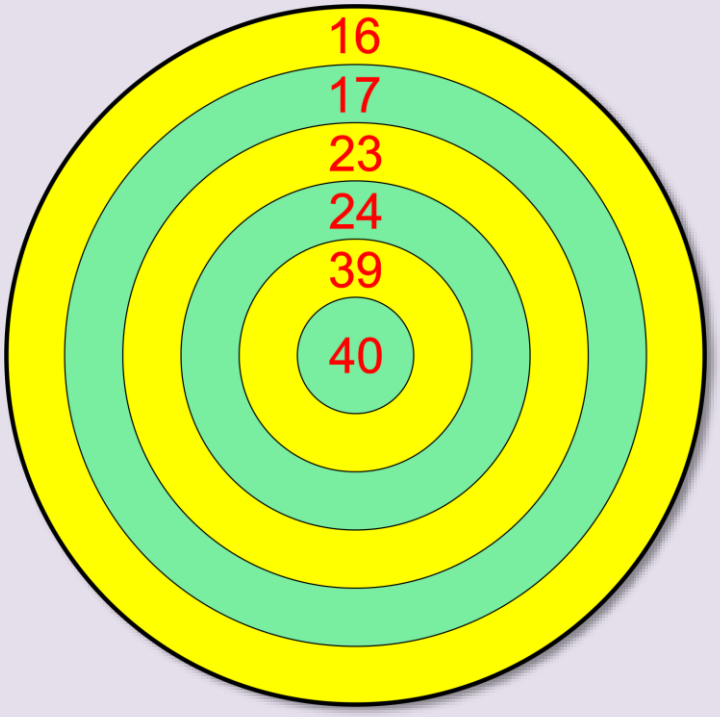
В задаче из немецкой книги *Gehirnjogging* речь идёт о стрельбе из лука по мишени:

Задача 26. Bogenschießen. Вот её условие на немецком языке:

Lilly hat im Garten eine Zielscheibe aufgebaut. ***Welche Felder muss sie treffen, um genau auf 100 Punkte zu kommen?***
(Sie darf beliebig viele Pfeile abschießen.)

Стрельба из лука

У Лили в саду висит мишень:



Как Лиля может выбить ровно 100 очков?
(Количество стрел может быть любым.)

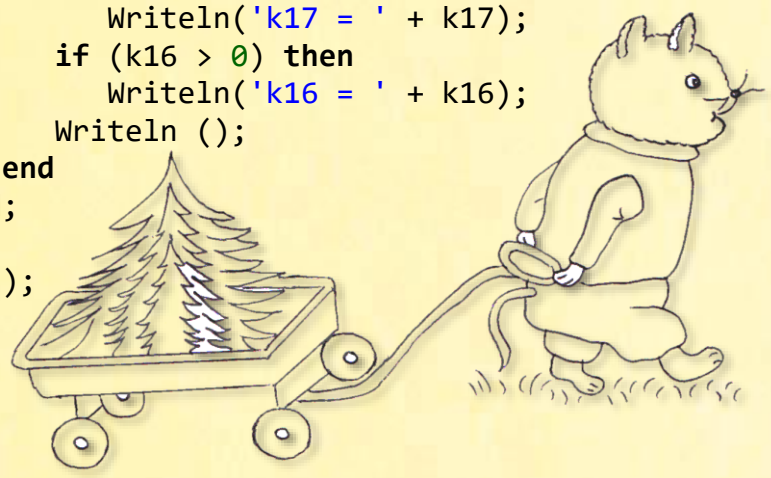
Компьютер не переносит немецкую букву ß (*эсцет*), поэтому название проекта пришлось изменить.

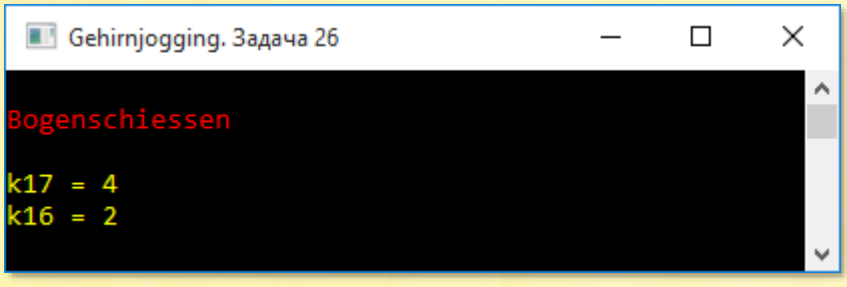
На оригинальной мишени число 24 встречается дважды, что уже совсем нехорошо. Пришлось заменить его числом 23.

Задача решается простым перебором во вложенных циклах *for*:

```
uses
   System;

// Gehirnjogging. Задача 26

const
   SCORE = 100;
   C40 = SCORE div 40;
   C39 = SCORE div 39;
   C24 = SCORE div 24;
   C23 = SCORE div 23;
   C17 = SCORE div 17;
   C16 = SCORE div 16;

// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

   for var k40 := 0 to C40 do
      for var k39 := 0 to C39 do
         for var k24 := 0 to C24 do
            for var k23 := 0 to C23 do
               for var k17 := 0 to C17 do
                  for var k16 := 0 to C16 do
                  begin
                     if (k40 * 40 + k39 * 39 +
                                 k24 * 24 + k23 * 23 +
                                 k17 * 17 + k16 * 16 = SCORE) then
                     begin
                        if (k40 > 0) then
                           Writeln('k40 = ' + k40);
                        if (k39 > 0) then
                           Writeln('k39 = ' + k39);
```

```
                        if (k24 > 0) then
                           Writeln('k24 = ' + k24);
                        if (k23 > 0) then
                           Writeln('k23 = ' + k23);
                        if (k17 > 0) then
                           Writeln('k17 = ' + k17);
                        if (k16 > 0) then
                           Writeln('k16 = ' + k16);
                        Writeln ();
                     end
                  end;

   Console.WriteLine();
end;


begin
   // заголовок окна:
   Console.Title := 'Gehirnjogging. Задача 26 ';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Bogenschiessen');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```





Если наверняка знать, что ответ единственный, то задачу можно решить и в уме.

Город

В журнале *Квантик*, №1 за 2016 год опубликованы задачи Первого тура математического конкурса. Мы решим **первую задачу**:

1. Город разделён рекой на две половины, в каждой половине живёт по миллиону человек. В первый год 2015 человек переселились из левой половины в правую; во второй год 2016 человек переселились из правой половины в левую; в третий год опять 2015 человек переселились слева направо; в четвёртый год – 2016 человек переселились справа налево, и так далее.

Докажите, что в какой-то год в каждой из половин снова окажется по миллиону жителей. Через сколько лет это случится?

«Докажите» здесь значит «решите задачу». Задача очень простая, но не сразу это сообразишь, поэтому мы напишем короткую программу, которая выдаст нам правильный ответ, который покажет, как можно решить эту задачу исключительно в уме.

Процедура **Solve** начинается с объявления **переменных**, известных вам по условию задачи:

```
uses
   System;

// Квантик, 2016, №1. Задача 1

// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
    Console.ForegroundColor := ConsoleColor.Yellow;

    // население:
    var gorod1 := 1000000;
    var gorod2 := 1000000;
    // переселение:
    var iz1 := 2015;
    var iz2 := 2016;
```

А дальше мы моделируем процесс переселения населения с насиженных мест в другую часть города.

Поскольку число переселений нам не известно (иначе мы бы знали ответ на задачу), то цикл *while* будет бесконечным, а прервём мы его тогда, когда выполнится требование задачи: в каждой половине города вновь окажется по 1 миллиону жителей.

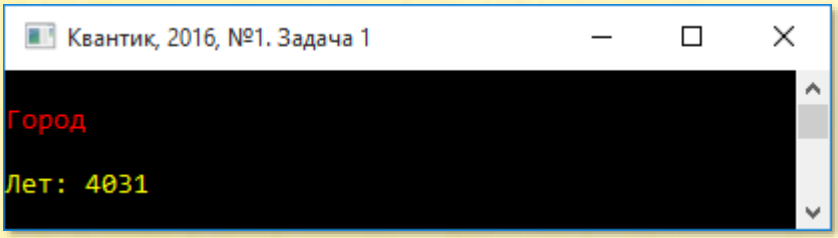
В первый год, а также во все последующие нечётные года iz1 человек переселяется из одной половины города в другую. Во второй год и во все чётные года, наоборот, из второй половины города переселяется iz2 человек в первую.

Через n лет население обеих половин города сравняется, и мы напечатаем ответ:

```
    // счётчик лет:
    var n := 0;
    while (true) do
    begin
        n += 1;
        if (n mod 2 = 1) then // из первого во второй
        begin
            gorod1 -= iz1;
            gorod2 += iz1;
        end
        else // из второго в первый
        begin
            gorod1 += iz2;
            gorod2 -= iz2;
        end;

        // проверяем численность:
        if (gorod1 = 1000000) then
        begin
            Writeln ('Лет: ' + n);
            break;
        end;
    end;

  Console.WriteLine();
end;
```

```
begin
   // заголовок окна:
   Console.Title := 'Квантик, 2016, №1. Задача 1';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Город');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

Запускаем программу и тут же узнаём, что долгодолгодолгожданное событие наступит через 4031 год:



Если бы найти такой город, в котором люди живут по 4000 лет, то можно было бы и попереселяться ежегодно.

Понятно, что эта задача совершенно искусственная, а число переселенцев просто обозначает предыдущий и текущий года.

Ответ 4031 нетрудно получить так: 2015 x 2 + 1. Откуда и следует элементарное решение.

Каждые 2 года население правой части города уменьшается на 1 человека. Но при этом из левой части города каждый нечётный год к ним добавляются 2015 человек. Когда в правой части останется 1000000 – 2015 человек, на следующий год в него прибудут 2015 человек из левой части города, и население возрастёт до 1000000 человек. Это будет как раз в тот год, о котором и спрашивается в условии задачи.

Как мы уже выяснили, население правой части города уменьшается на 1 человека за 2 года, поэтому на 2015 человек население уменьшится за 2015 x 2 = 4030 лет. А на следующий, 4031-й год 2015 человек из левой части города увеличат его до 1 миллиона.

Хорошее дело – программирование, но иногда и мозгами нелишне пораскинуть!

Наименьшее число

В журнале *Наука и жизнь*, №2 за 1968 год, на странице 57 напечатана такая задача:

НАЙТИ НАИМЕНЬШЕЕ ЧИСЛО

Найдите наименьшее число, которое при делении на 2, 3, 4, 5 и 6 дает остатки соответственно 1, 2, 3, 4 и 5.

Задача исключительно на сообразительность. Вот журнальное решение задачи:

НАЙТИ НАИМЕНЬШЕЕ ЧИСЛО

Прибавим к искомому числу 1. Полученная сумма должна делиться одновременно на 2, 3, 4, 5 и 6. Наименьшее такое число 60. Следовательно, искомое число:

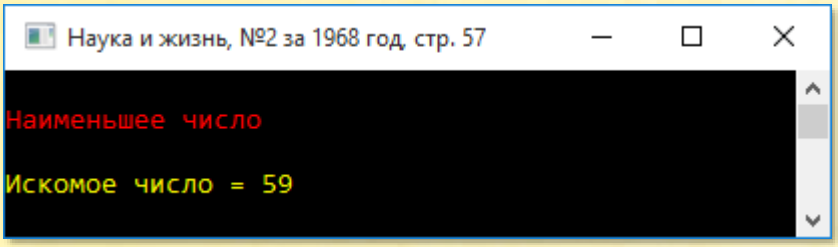
60 — 1 = 59.

Не каждый этим похвальным даром обладает, поэтому мы пишем простенькую программу:

```
uses
   System;


// Наука и жизнь, №2 за 1968 год, стр. 57

// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

    var n := 0;
    while (true) do
    begin
        n += 1;
        var flg := true;
        for var i := 2 to 6 do
        begin
            if (n mod i <> i - 1) then
            begin
                flg := false;
                break;
            end;
        end;
        if (flg) then
        begin
             Writeln ('Искомое число = ' + n);
             break;
        end;
    end;

   Console.WriteLine();
end;


begin
   // заголовок окна:
   Console.Title := 'Наука и жизнь, №2 за 1968 год, стр. 57';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Наименьшее число');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();
```

```
    Solve();

    Console.WriteLine();
    Console.ForegroundColor := ConsoleColor.Red;
end.
```

Когда искомое число заведомо небольшое, то его легко найти простым перебором в бесконечном цикле ***while***. Он заканчивается, когда найдено первое число, удовлетворяющее всем условиям задачи:

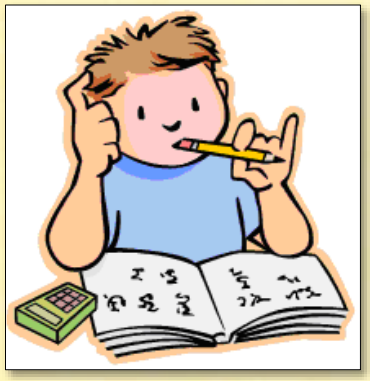


Вопреки ожиданиям наш ответ полностью совпал с «сообразительным» …

Трёхзначное число

В книге *600 задач на сообразительность*, на странице 112 напечатана **задача 41**:

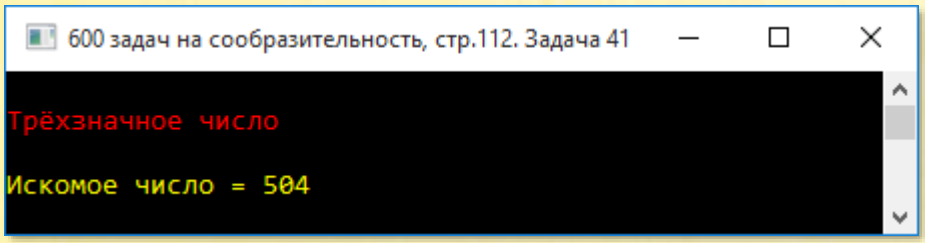
> ***Трёхзначное число***
> *Если от трёхзначного числа отнять 7, то оно разделится на 7; если отнять от него 8, то оно разделится на 8; если отнять от него 9, то оно разделится на 9. Какое это число?*



Опять задача на сообразительность, которой мы как бы не обладаем, а потому сразу садимся за компьютер и решаем задачу:

```
uses
   System;

// 600 задач на сообразительность, стр.112. Задача 41

// РЕШАЕМ ЗАДАЧУ
procedure Solve();
begin
   Console.ForegroundColor := ConsoleColor.Yellow;

    var n := 0;
    while (true) do
    begin
        n += 1;
        if  ((n - 7) mod 7 <> 0) then
            continue;
        if ((n - 8) mod 8 <> 0) then
            continue;
        if ((n - 9) mod 9 <> 0) then
            continue;
        break;
    end;
    Writeln ('Искомое число = ' + n);

   Console.WriteLine();
end;


begin
   // заголовок окна:
   Console.Title := '600 задач на сообразительность, стр.112. Задача
41';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Трёхзначное число');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   Solve();

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```

Ответ такой:



Он, безусловно, правильный, но, чтобы его получить, вполне можно обойтись и смекалкой.

Действительно, если число $n - 7$ нацело делится на 7, то и число n кратно 7. Аналогично для чисел 8 и 9. Стало быть искомое число одновременно делится на 7, 8 и 9.

Минимальное число, обладающее такими свойствами, равно:

7 * 8 * 9 = 504

Следующее такое число 504 * 2 = 1008, но оно не трёхзначное.

Треугольные числа

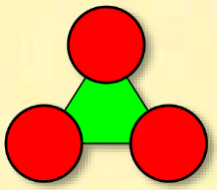
Треугольные числа – это частный вид *фигурных чисел*.

Если из определённого числа кружков можно построить правильный треугольник, то такое число и называют треугольным.

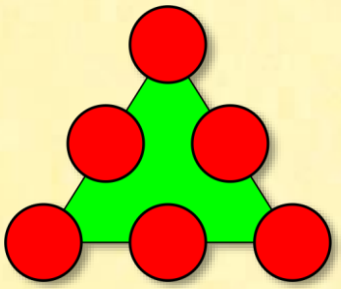
Самое первое треугольное число – это единица. Правда, один кружок не очень-то похож на треугольник, но у математиков свой взгляд на вещи.

Следует добавить, что **нулевое** треугольное число представляет собой треугольник, которого вообще не видно. Он существует только в мозгу математиков.

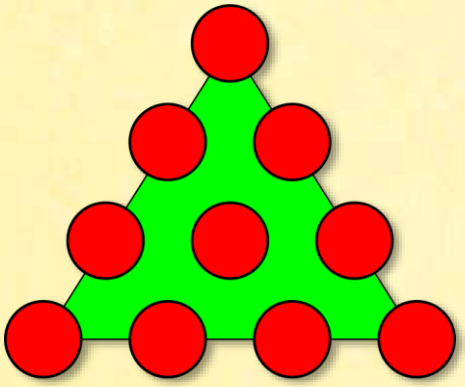
Второе треугольное число – тройка. Из трёх кружочков уже можно построить вполне реалистичный треугольник:



Третье треугольное число – шестёрка:



Четвёртое треугольное число – десятка:



Дальше можно не продолжать – закономерность понятна:

Первое число равно 1.

Второе – 1 + 2.
Третье - 1 + 2 + 3.
Четвёртое - 1 + 2 + 3 + 4.
Пятое - 1 + 2 + 3 + 4 + 5.

n-ное треугольное число равно сумме первых n натуральных чисел.

Ряд треугольных чисел представляет собой простейшую арифметическую прогрессию, поэтому любое треугольное число можно легко вычислить по формуле:

$$T_n = ½ n(n + 1) \quad (1)$$

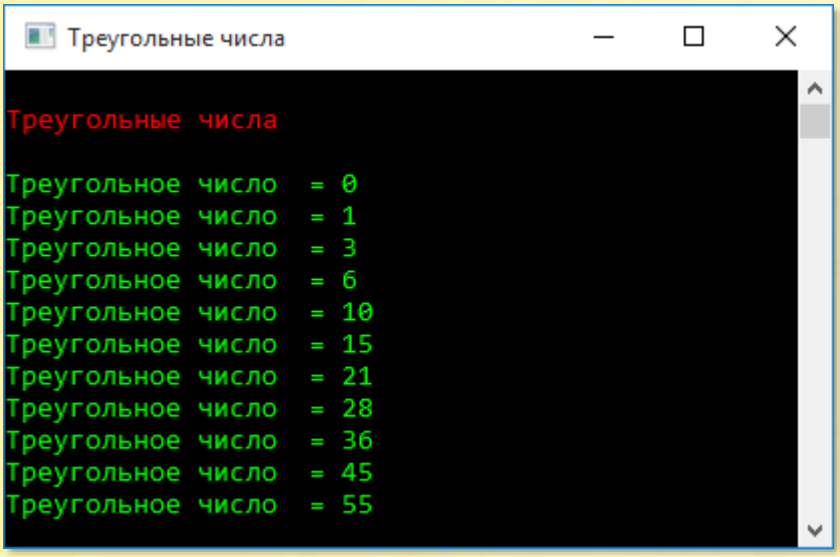
Однако из наших нарисованных треугольников видно, что второе треугольное число получается, если к первому прибавить 2. Третье – если ко второму прибавить 3. Для любого треугольного числа верна такая рекуррентная формула:

$$T_n = T_{n-1} + n$$

Из неё хорошо видна прямая аналогия с рекурсивным вычислением факториалов. Нам нужно только заменить умножение сложением:

```
uses
   System;

type
   int = integer;
   bool = boolean;

// Треугольные числа


// Рекурсивный способ вычисления треугольных чисел

// РЕКУРСИВНАЯ ФУНКЦИЯ
function tri(n: int): int;
begin
    if (n = 0) then
    begin
        Result := 0;
```

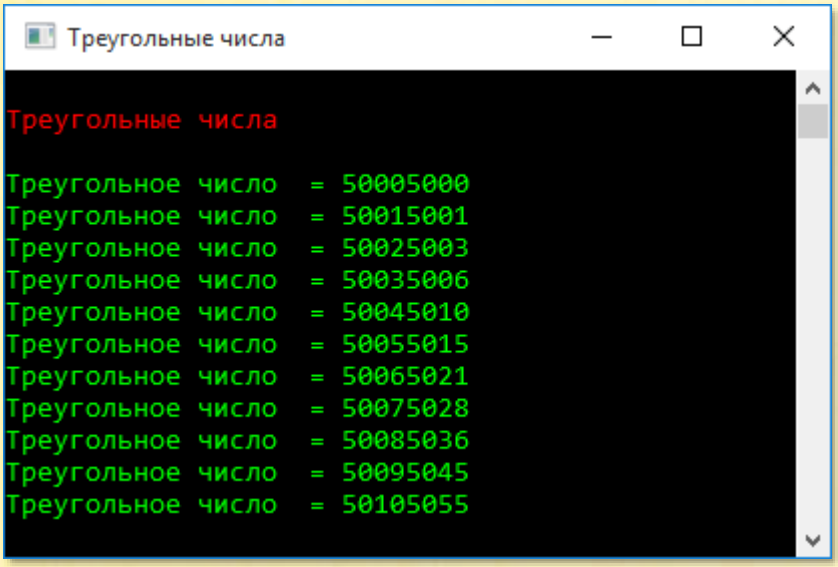
```
        exit;
    end;
    Result := n + tri(n-1);
end;


begin
   // заголовок окна:
   Console.Title := 'Треугольные числа';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Треугольные числа');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

   for var n := 0 to 10 do
        Writeln ('Треугольное число ' + ' = ' + tri(n));

   Console.WriteLine();
   Console.ForegroundColor := ConsoleColor.Red;
end.
```



Если вам вдруг понадобятся очень большие треугольные числа, то их можно получить от нерекурсивной функции:

```
// НЕРЕКУРСИВНАЯ ФУНКЦИЯ
function tri2(n : int): BigInteger;
begin
```

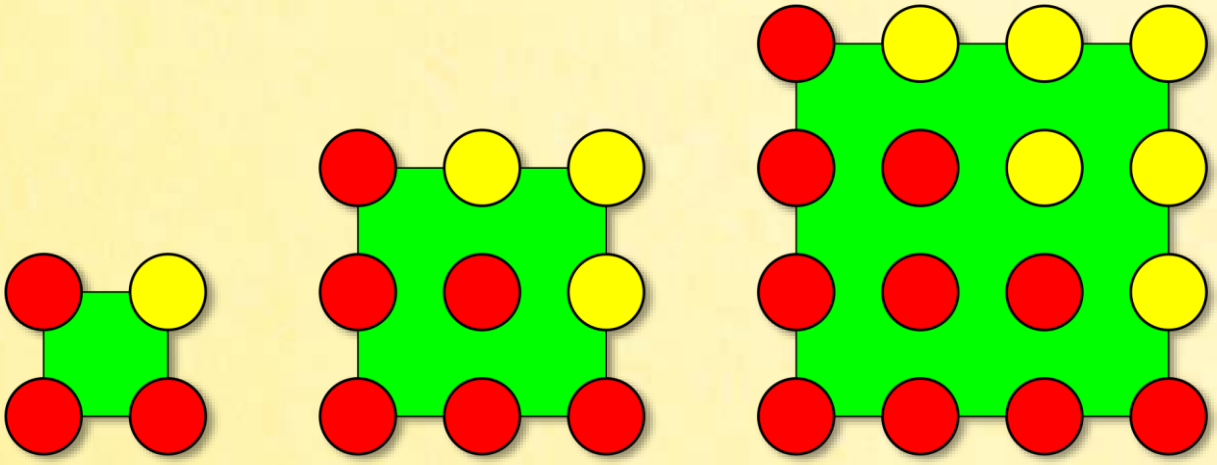
```
    if (n <= 1) then
    begin
        Result := n;
        exit;
    end;

    Result := n * (n + 1) div 2;
end;
```



Интересно, что сумма двух последовательных треугольных чисел - это **квадратное число**:

$$T_n + T_{n+1} = (n+1)^2$$

На рисунке это свойство хорошо видно:

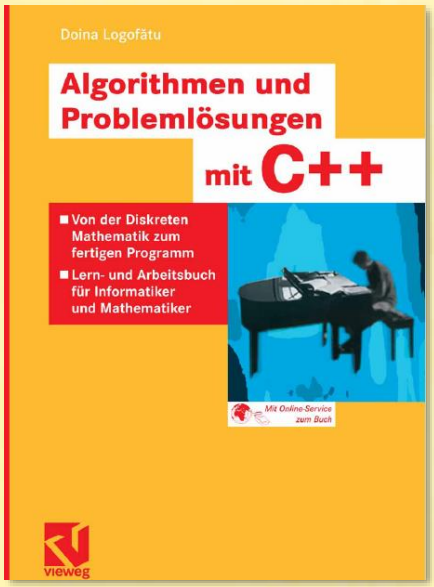


А точное доказательство этого утверждения легко получить из формулы (1), если найти сумму чисел T_n и T_{n+1}.

Менее очевидно такое свойство треугольных чисел: *любое целое неотрицательное число можно представить в виде суммы не более трёх треугольных чисел*. Это предположение высказал Пьер Ферма в 1638 году, а доказал его в 1796 году Карл Гаусс.

Великолепная четвёрка

В книге *Algorithmen und Problemlösungen mit C++,* на страницах 300-302 решается такая **задача**.



Их четвёрки с помощью трёх простых операций получить любое натуральное число.

Операции такие:

- добавить 4 в конец
- добавить 0 в конец
- разделить число на 2, если оно чётное

Эту задачу проще решить **с конца**, то есть заданное число превратить в четвёрку. Тогда и операции станут противоположными:

- удалить 4, если она стоит в конце числа
- удалить 0, если он стоит в конце числа
- умножить число на 2

В **главном блоке** мы вводим любое целое число. Если это нуль или отрицательное число, то программа закрывается. В противном случае число *num* передаётся функции **solve**. Кроме того, она получает *сообщения*, которые поясняют действие функции.

Если число *num* - **четвёрка** (основной случай), то решение заканчивается. В противном случае мы рекурсивно выполняем операции над текущим числом *num*.

Если число *num* **заканчивается на нуль**, то мы удаляем его. Для этого достаточно разделить число на 10.

Аналогично мы поступаем, если число **заканчивается на четвёрку**.

Если число **заканчивается на другие цифры**, то мы умножаем его на 2.

Эти действия продолжаем до тех пор, пока число не обратится в четвёрку. Тогда мы печатаем все действия с комментариями в обратном порядке – по мере возврата из вызванных функций.

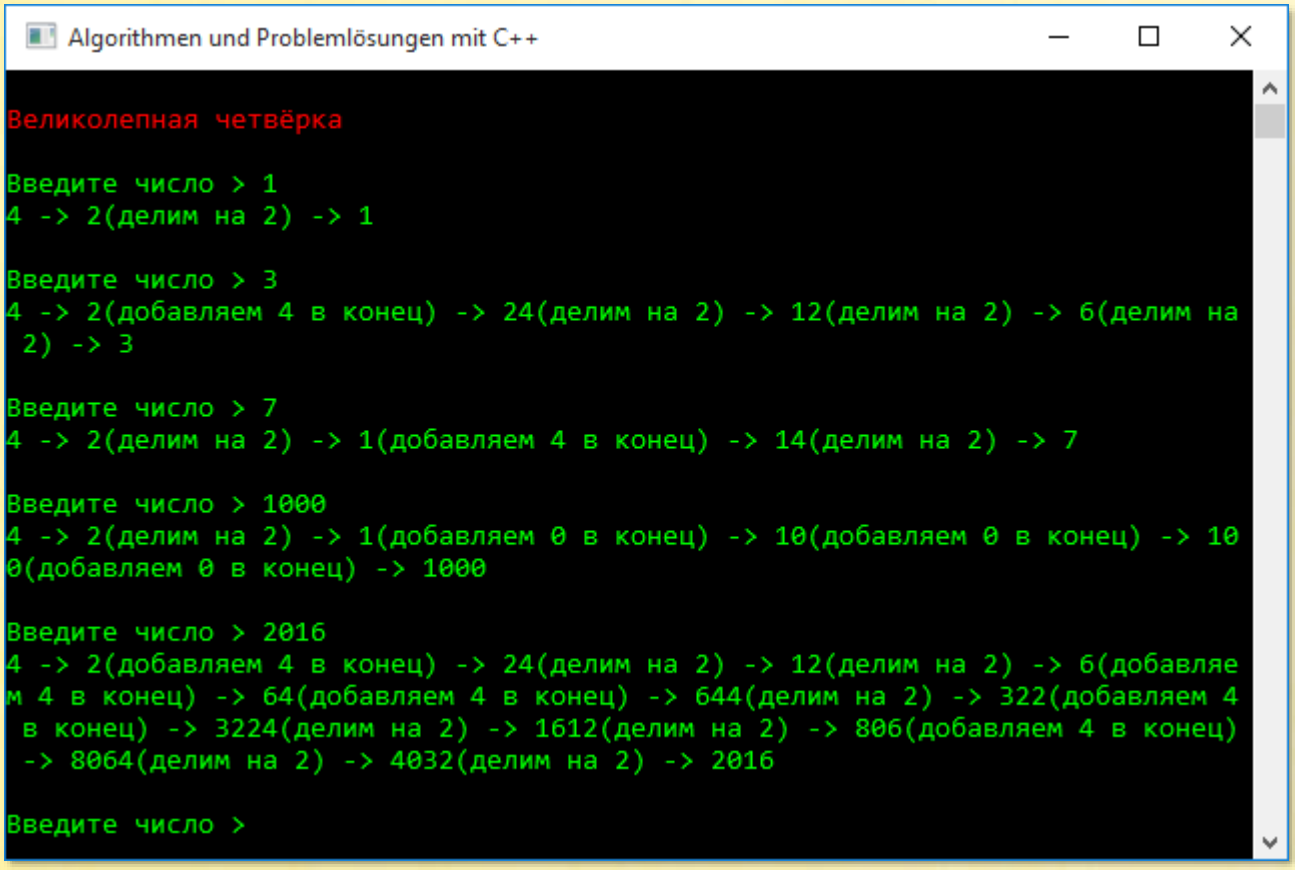
Обратите внимание, что в функции **solve** операции противоположные, а комментарии «прямые», так как функция печатает все действия в прямом порядке.

```
uses
   System;

type
   int = integer;
   bool = boolean;

// Algorithmen und Problemlosungen mit C++, стр. 300-302
```

```
// РЕШАЕМ ЗАДАЧУ
procedure solve(num: int; message: string);
begin
   if (num = 4) then
   begin
      Write(num);
      exit;
   end;

   // последняя цифра числа:
   var lastDig := num mod 10;
   // добавляем 0 в конец:
   if (lastDig = 0) then
      solve(num div 10, '(добавляем 0 в конец)')
   // добавляем 4 в конец:
   else if (lastDig = 4) then
      solve(num div 10, '(добавляем 4 в конец)')
   // делим на 2:
   else
      solve(num * 2, '(делим на 2)');

   Write(' -> ' + num + message);
end;


begin
   // заголовок окна:
   Console.Title := 'Algorithmen und Problemlosungen mit C++';
   Console.WriteLine('');
   Console.ForegroundColor := ConsoleColor.Red;
   Console.WriteLine('Великолепная четвёрка');
   Console.ForegroundColor := ConsoleColor.Green;
   Console.WriteLine();

    // бесконечный цикл ввода данных -
    // пока пользователь не закроет программу
    // или не введёт нуль или отрицательное число:
   while (true) do
   begin
      Write('Введите число > ');
```

```
        var num := 0;
        Readln(num);
        // если пользователь ввёл 0 или отрицательное число,
        // то программу закрываем:
        if (num <= 0) then
           exit;
        // решаем задачу:
        solve(num, '');
        Writeln ();
        Writeln ();
    end;

    Console.WriteLine();
    Console.ForegroundColor := ConsoleColor.Red;
end.
```

На рисунке вы видите, как работает наша программа:



Рекурсивный тортик

В книге **Математические изюминки [XP92]**, на страницах 9-11 Росс Хонсбергер показывает, как решить такую задачу:

> Расположим *n* точек на окружности и соединим их попарно хордами. Предположим, что никакие три хорды не имеют общей точки внутри круга.
>
> **На сколько областей разобьётся круг этими хордами?**

Вывод формулы вы можете проследить по книге, нас же интересует только **формула**:

$$NR(n) = C_n^2 + C_n^4 + 1$$

Здесь:

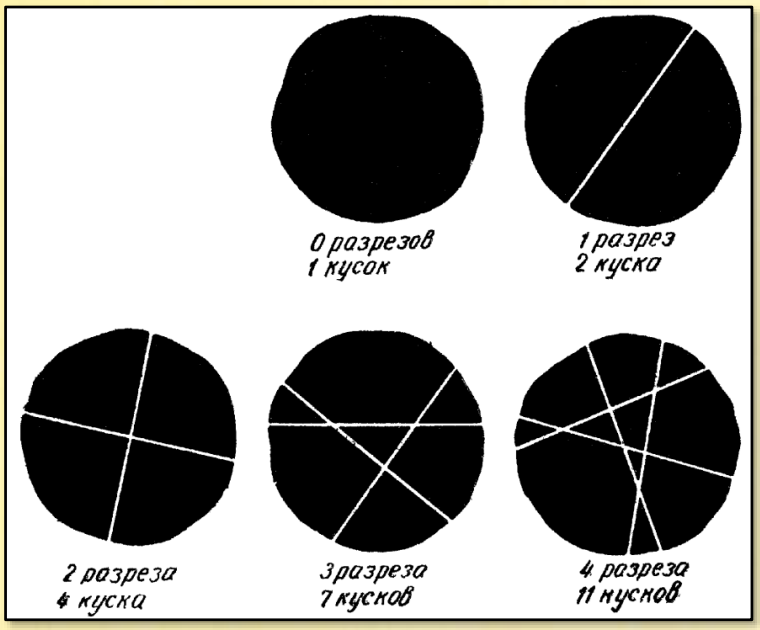
- NR – число областей, на которые хорды разбивают круг
- n – число точек на окружности
- C_n^2 и C_n^4 - число сочетаний по 2 и по 4 из *n*

Подобную задачу решает и Мартин Гарднер в книге **Математические досуги [ГМ72]**, на страницах 82-87.

Она облечена в более занимательную форму:

> **На какое максимальное число кусков** можно разделить круглый пирог, если сделать *n* разрезов, каждый из которых пересекает все остальные?

На рисунке вы можете внимательно проследить судьбу тортика, подвергнутого означенным разрезам.



По этим данным с помощью *метода конечных разностей* легко выводится **формула** для подсчёта кусков:

$$NR = \frac{1}{2} n^2 + \frac{1}{2} n + 1 = \frac{1}{2} n(n + 1) + 1 \qquad (1)$$

Существует и **рекуррентная формула**:

$$NR(0) = 1$$
$$NR(n) = n + NR(n-1) \text{ при } n > 0$$

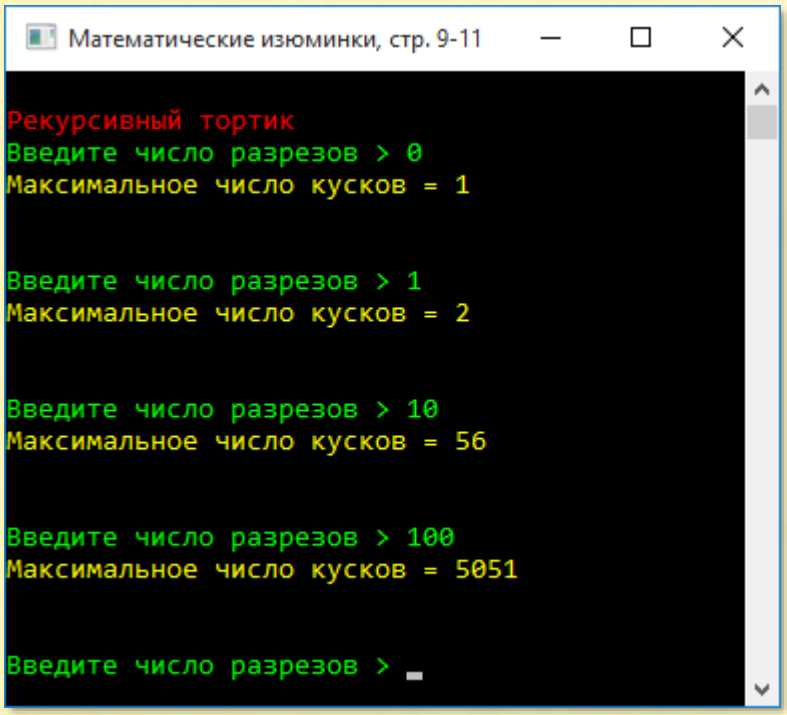
Давайте в помощь домохозяйкам и кондитерам напишем программу, умеющую подсчитывать куски торта, которые получаются при заданном числе разрезов.

Поскольку в реальной жизни вряд ли придётся разрезать даже самый большой торт на тысячи частей (то есть в пыль!), то для практических нужд вполне достаточно **рекуррентной** формулы:

```
uses
   System;
```

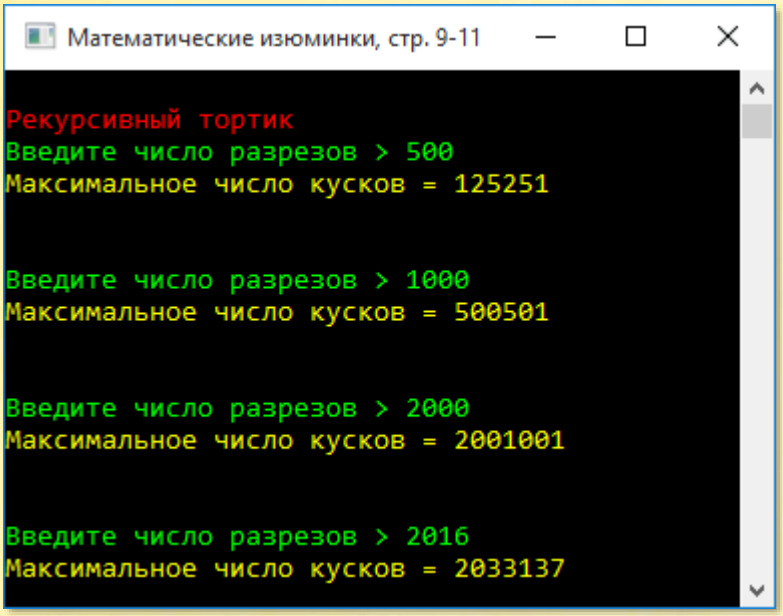
```
type
    int = integer;
    bool = boolean;

// Математические изюминки, стр. 9-11

// РЕЖЕМ ТОРТИК РЕКУРСИВНО
function TortikRec(num: int): int;
begin
    Console.ForegroundColor := ConsoleColor.Yellow;

    if(num < 1) then
    begin
        Result := 1;
        exit;
    end
    else
        Result := num + TortikRec(num - 1);
end;


begin
    // заголовок окна:
    Console.Title := 'Математические изюминки, стр. 9-11';
    Console.WriteLine('');
    Console.ForegroundColor := ConsoleColor.Red;
    Console.WriteLine('Рекурсивный тортик');

    // бесконечный цикл ввода данных -
    // пока пользователь не закроет программу
    // или не введёт нуль или отрицательное число:
    while (true) do
    begin
        Console.ForegroundColor := ConsoleColor.Green;
        Write('Введите число разрезов > ');
        var num := 0;
        Readln(num);
        // если пользователь ввёл отрицательное число,
        // то программу закрываем:
        if (num < 0) then
            exit;
```

```
        // находим число разрезов:
        num := TortikRec(num);
        // печатаем число кусков:
        Writeln('Максимальное число кусков = ' + num);

        Writeln ();
        Writeln ();
    end;
end.
```



Но если вас интересуют и теоретические изыскания в кондитерской науке, то нужно прибегнуть к формуле (1):

```
// РЕЖЕМ ТОРТИК НЕ РЕКУРСИВНО
function Tortik(num: int): int;
begin
    Result := num*(num + 1) div 2 + 1;
end;
```



Тортик

Впрочем, хозяйке гораздо интереснее и полезнее знать, сколько нужно сделать разрезов, чтобы получить вполне определённое число кусков торта.

В главном блоке она задаёт нужное число кусков и получает от функции *Tortik* необходимое число разрезов:

```
uses
   System;

type
    int = integer;
    bool = boolean;

// Математические изюминки, стр. 9-11

begin
    // заголовок окна:
    Console.Title := 'Математические изюминки, стр. 9-11';
    Console.WriteLine('');
    Console.ForegroundColor := ConsoleColor.Red;
```

```
    Console.WriteLine('Тортик');

    // бесконечный цикл ввода данных -
    // пока пользователь не закроет программу
    // или не введёт нуль или отрицательное число:
    while (true) do
    begin
        Console.ForegroundColor := ConsoleColor.Green;
        Write('Введите число кусков > ');
        var num := 0;
        Readln(num);
        // если пользователь ввёл отрицательное число,
        // то программу закрываем:
        if (num < 0) then
            exit;

        // находим число разрезов:
        var tile := Tortik(num);

        // печатаем число разрезов:
        Console.ForegroundColor := ConsoleColor.Yellow;
        Writeln('Минимальное число разрезов = ' + tile);
        var unwanted := tile * (tile + 1) div 2 + 1 - num;
        Writeln('Число лишних кусков = ' + unwanted);

        Writeln ();
        Writeln ();
    end;
end.
```

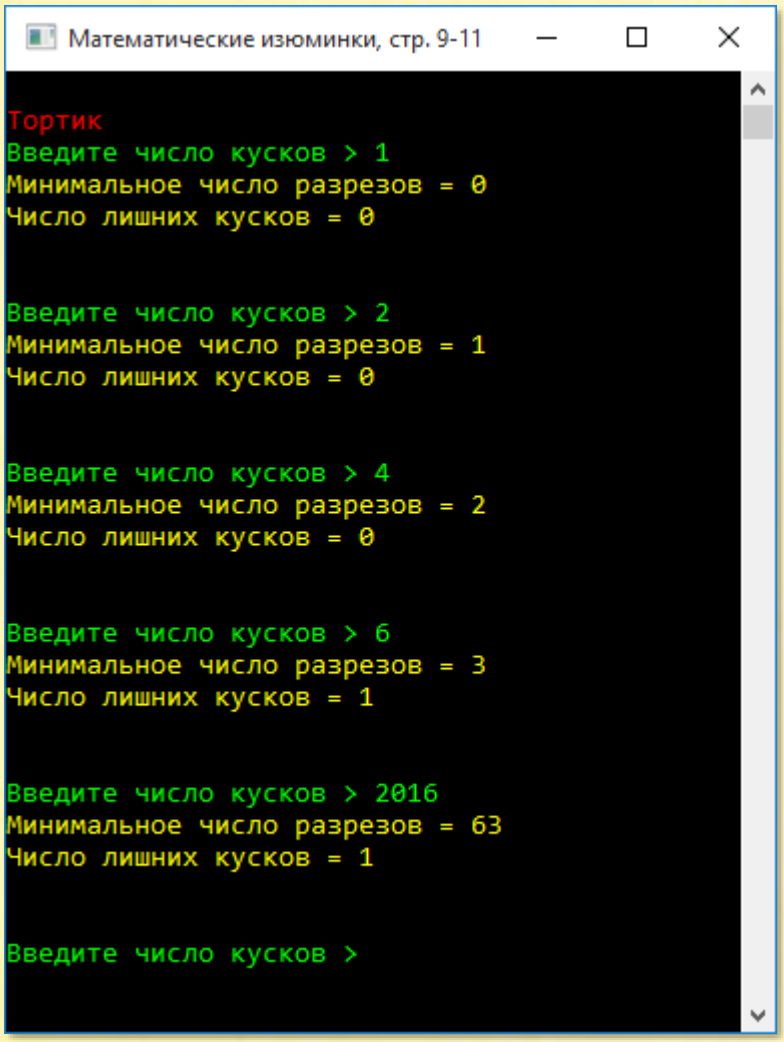
В функции Tortik нам удобнее пользоваться *нерекуррентной* формулой.

Она получает необходимое число кусков n и начинает последовательно делать *num* разрезов – до тех пор, пока число кусков не сравняется или не превысит заданного числа кусков:

```
// РЕЖЕМ ТОРТИК
function Tortik(num: int): int;
begin
```

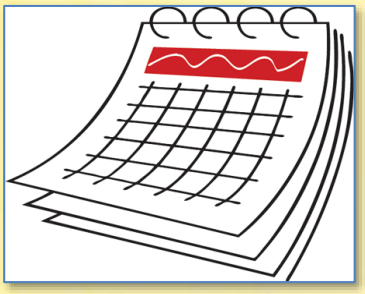
```
    Result := 0;
    while(num > (Result*(Result + 1) div 2 + 1)) do
        Result += 1;
end;
```

На рисунке вы видите работу этой во всех отношениях полезной для дома и для семьи программы:



В некоторых случаях остаются «лишние» куски. Вот почему так важно знать математику и **лично** нарезать тортики кусками!

Столетие



В кашне, ладонью заслонясь,
Сквозь фортку крикну детворе:
Какое, милые, у нас
Тысячелетье на дворе?

Борис Леонидович Пастернак, *Про эти стихи*

Первая задача школьного этапа Всероссийской олимпиады по информатике 2016-2017 для 7-8 классов:

> *По четырёхзначному номеру года, запрошенного с клавиатуры, определите номер столетия (например, для 1342 года ответ XIV век, для 1918 года – XX век). Учесть, что началом века считается первый, а не нулевой год. (То есть, 2000-й год – это последний год XX века).*

Так как век – это 100 лет, то номер года нужно разделить на 100. Проверим, что получается для 20-го века:

1901 / 100 = 19.01
2000 / 100 = 20.0

Вывод 1: в частном нужно отбрасывать дробную часть.

Недаром **в программировании счёт начинают с нуля**. Мы получили для начала века другой результат, чем для всех остальных именно потому, что люди считают с единицы, а не с нуля. Вычтем из числа года 1, тогда в 20 веке года попадут в диапазон 1900..1999. При делении нацело получим:

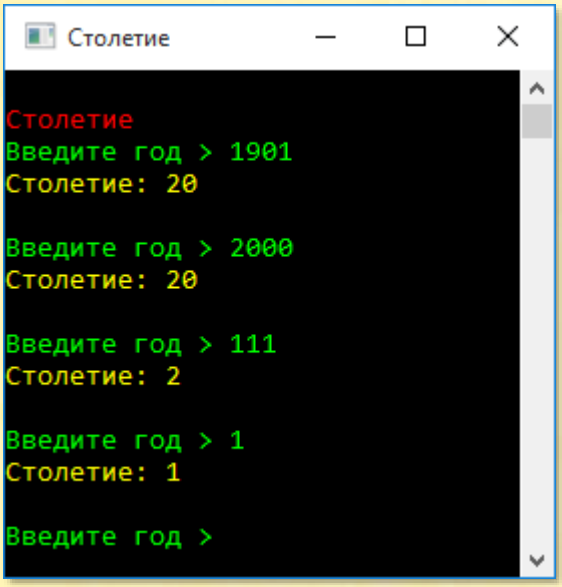
1900 / 100 = 19
1999 / 100 = 19

Все промежуточные года также дадут частое, равное 19. Уже хорошо, но век получается на 1 меньше, чем следует. Добавим к частному 1 – и задача решена.

Программа в бесконечном цикле *while* спрашивает у пользователя год, а в переменную vek записывает номер столетия:

```
uses
   System;

// Столетие

begin
    // заголовок окна:
    Console.Title := 'Столетие';
    Console.WriteLine('');
    Console.ForegroundColor := ConsoleColor.Red;
    Console.WriteLine('Столетие');

    // бесконечный цикл ввода данных -
    // пока пользователь не закроет программу
    // или не введёт нуль или отрицательное число:
    while (true) do
    begin
        Console.ForegroundColor := ConsoleColor.Green;
        Write('Введите год > ');
        var god := 0;
        Readln(god);
        // если пользователь ввёл 0 или отрицательное число,
        // то программу закрываем:
        if (god <= 0) then
            exit;

        // находим столетие:
        var vek := (god - 1) div 100 + 1;
        Console.ForegroundColor := ConsoleColor.Yellow;
        // печатаем столетие:
        Writeln('Столетие: ' + vek);

        Writeln ();
    end;
end.
```

Как и полагается, проверяем программу на экстремальных значениях:



Очевидно, что все промежуточные года будут вычислены верно. То же самое относится и к другим векам. Почему номер года должен быть четырёхзначным, непонятно. Для всех натуральных чисел программа вычисляет столетие верно.

Суперпростые числа

Под суперпростыми понимают **разные** числа, но мы исследуем только такие числа, которые и целиком – простые, и все их дробные части – также простые. Для примера возьмём простое число **1373**. Разобьём его всеми возможными способами на две части:

```
1-373
13-73
137-3
```

Оказывается, что все «дробные числ»а – *1, 3, 13, 73, 137, 373* - также простые. А раз это так, то исходное число *1373* мы по полному праву можем именовать **суперпростым**!

Наша **задача** - отыскать все суперпростые числа заданной длины *len*. Так как однозначные числа нельзя поделить на части, то они формально суперпростые, но очень неинтересные, поэтому мы разрешим пользователю задавать длину чисел в разумном диапазоне 2..7:

```
uses
   System;

type
    int = integer;
    bool = boolean;

// Суперпростые числа

begin
    // заголовок окна:
    Console.Title := 'Суперпростые числа';
    Console.WriteLine('');
    Console.ForegroundColor := ConsoleColor.Red;
    Console.WriteLine('Суперпростые числа');

    // бесконечный цикл ввода данных -
    // пока пользователь не закроет программу
    // или не введёт нуль или отрицательное число
    var len := 0;
    while(true) do
        repeat
            Writeln ();
            Console.ForegroundColor := ConsoleColor.Green;
            Write('Введите число 2..7 > ');
            Readln(len);
            // если пользователь ввёл 0 или отрицательное число,
            // то программу закрываем:
            if (len <= 0) then
                exit;
            Solve(len);
        until ((len >= 2) and (len <= 7));
    Writeln ();
end.
```

По заданной длине чисел мы находим максимальное число *max*, которое на 1 больше максимального из заданных:

```
procedure Solve(len: int);
begin
    Writeln ();
    // ищем суперпростые числа
    // заданной длины:
    var max := 1;
    for var i := 1 to len do
        max *= 10;
```

Например, для len = 2 мы получим 1 с *двумя* нулями – *100*. *Наибольшее двузначное* число на 1 меньше наименьшего трёхзначного, то есть *99*.

Теперь в цикле ***for*** мы перебираем все числа заданной длины. Например, для двузначных чисел переменная цикла *num* изменяется от 100 : 10 = *10* до *99*:

```
for var num := max div 10 to max - 1 do
begin
    if (not Prime(num)) then
        continue;
```

Поскольку исходное, длинное число должно быть простым, то и его необходимо проверить, и если оно не соответствует предъявленным требованием, цикл переходит к проверке *следующего* числа.

Тут нам необходима функция для проверки заданного числа на простоту:

```
function Prime(number: int): bool;
begin
    // проверяем, делится ли введенное число на числа
    // 2..корень квадратный из числа;
    for var i := 2 to trunc(sqrt(number)) do
    begin
        if (number mod i = 0) then
        begin
            Result := false;
            exit;
        end;
    end;
    Result := true;
end;
```

Она возвращает значение *true*, если число *number* простое, и *false* – если составное.

Если длинное число *num* оказалось простым, то нам необходимо проверить и все его составные части. Для этого мы делим его на вспомогательную переменную *i*. При начальном значении, равном 10, мы разобьём число *num* на такие части:

```
1373 / 10 = 137
1373 % 10 = 3
```

В следующей итерации значение переменной *i* станет равным 10 x 10 = 100, и мы получим такие части исходного числа:

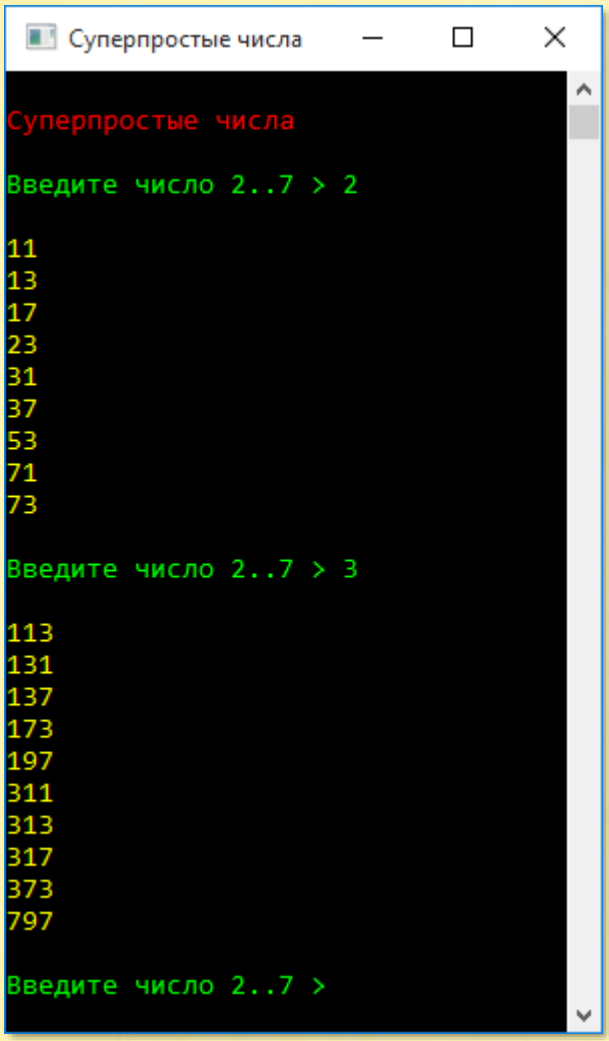
```
1373 / 100 = 13
1373 % 100 = 73
```

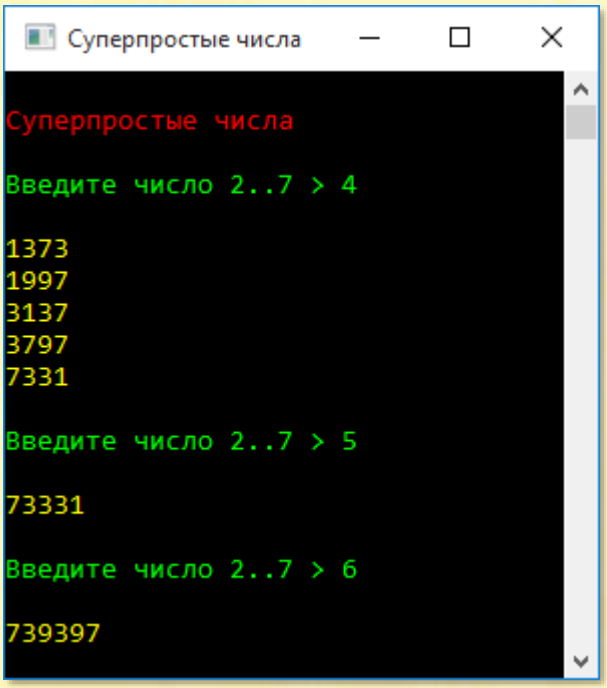
Вы легко продолжите этот процесс расщепления до самого конца. А мы должны убедиться, что каждая из частей также является простым числом. Для этого мы, естественно, опять вызываем функцию *Prime*. Если проверяемое число окажется составным (функция вернёт *false*), то следующие составные части проверять бесполезно, поэтому с помощью оператора *break* мы досрочно выходим из цикла *while*:

```
var i := 10;
while (i < max) do
begin
    if (not Prime(num div i)) then
        break;
    if (not Prime(num mod i)) then
        break;
    i *= 10;
end;
if (i = max) then
begin
    Console.ForegroundColor := ConsoleColor.Yellow;
    // печатаем суперпростое число:
```

```
            Writeln(num);
        end;
    end;
end;
```

На выходе (досрочном или «нормальном») мы сравниваем конечное значение переменной *i* с максимальным - *max*. Легко понять, что если они совпадут, то все составные части исходного числа оказались простыми, и нам ничего не остаётся, как только напечатать это число в *Консольном окне*.





Суперпростых чисел длины 7 не существует.

Числа Лишрел

Вот очень простой **196-алгоритм**:

1. Возьмите любое натуральное число num.
2. Найдите его сумму с обращённым числом *num = num + rev*.
3. Если сумма num – не палиндромическое число, то переходите к п.2.

Операцию 2 называют *Перевернуть и сложить* (*Reverse-Then-Add*).

Для большинства чисел это процесс заканчивается очень быстро, но некоторые числа порождают бесконечные циклы (или только очень длинные – это никому не известно). Наименьшее из таких упрямых чисел – **196**. Поэтому и алгоритм, и сама проблема включают это число в своё название.

Такие числа называют **числами Лишрел** (по-английски - *Lychrel numbers*). Это название придумал Уэйд Ван Лэндигэм (Wade VanLandingham), переставив буквы в имени своей подруги Cheryl и добавив лично от себя букву L).

Если число «упрямое», то сумма может быстро переполнить переменную типа *int*, поэтому, чтобы продвинуться дальше, этот тип следует заменить более ёмким типом **BigInteger**.

В **главном блоке** пользователь вводит число для проверки. Если оно меньше 1, то программа закрывается. Если оно изначально палиндромическое (например, 11 или 22), то мы печатаем сообщение об этом, и пользователь возвращается к вводу другого числа:

```
uses
   System;
```

```
type
    int = integer;
    bool = boolean;

// Числа Лишрел

begin
    // заголовок окна:
    Console.Title := 'Числа Лишрел';
    Console.WriteLine('');
    Console.ForegroundColor := ConsoleColor.Red;
    Console.WriteLine('Числа Лишрел');

    // бесконечный цикл ввода данных
    while (true) do
    begin
        Console.ForegroundColor := ConsoleColor.Green;
        Write('Введите число > ');
        var n := 0;
        Readln(n);
        // если пользователь ввёл число < 1,
        // то программу закрываем:
        if (n < 1) then
            exit;

        Console.ForegroundColor := ConsoleColor.Yellow;
        var num: BigInteger := n;
        if (IsPalindromicNumber(num)) then
        begin
            Writeln('Число ' + num.ToString() +
                    ' уже палиндромическое!');
            Writeln ();
            continue;
        end;
```

Если «криминала» не обнаружено, программа печатает исходное число и переходит к выполнению описанного выше алгоритма.

В переменной niter подсчитывается число шагов, потребовавшихся для получения палиндромического числа (не забывайте, что некоторые числа создают, вероятно, бесконечные последовательности; например число 196 после 2 415 836 итераций превращается в число с 1 миллионом цифр, которое всё ещё не стало палиндромом). По ходу нахождения обращённых чисел и сумм программа печатает их значения:

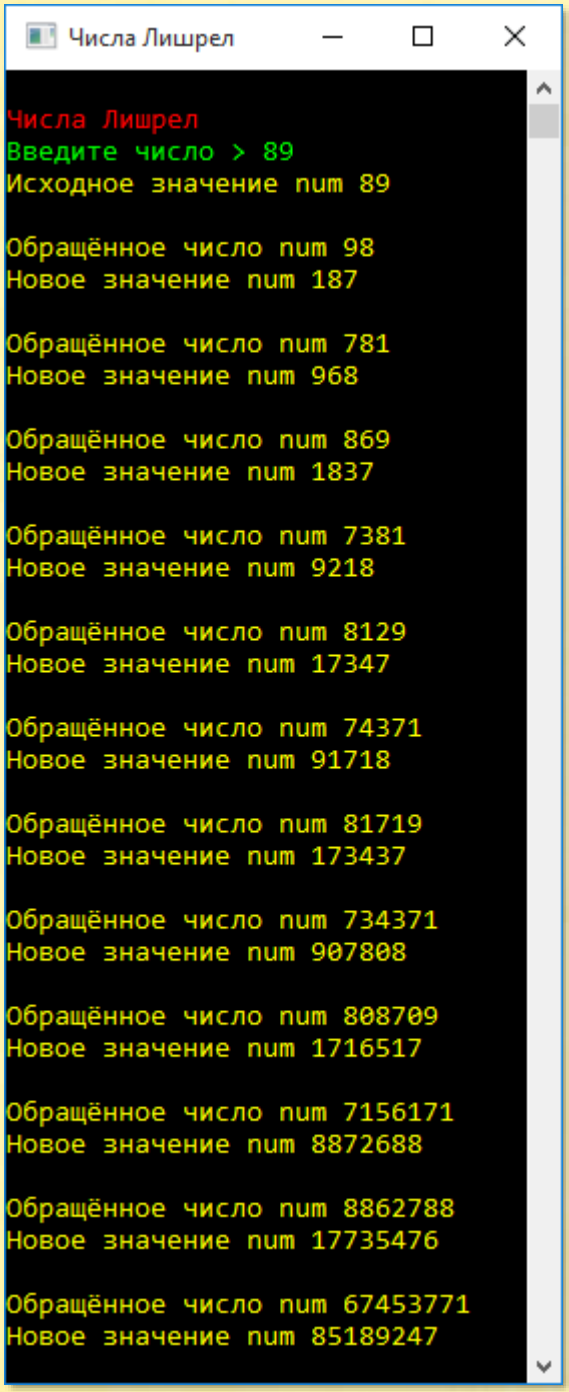
```
        Writeln('Исходное значение num ' + num.ToString());
        var niter := 0;
        repeat
            Writeln ();
            var rev := ReverseNum(num);
            Writeln('Обращённое число num ' + rev.ToString());
            num += rev;
            Writeln('Новое значение num ' + num.ToString());
            niter += 1;
        until (IsPalindromicNumber(num));

        Writeln('Число итераций = ' + niter);
        Writeln ();
    end;
end.
```

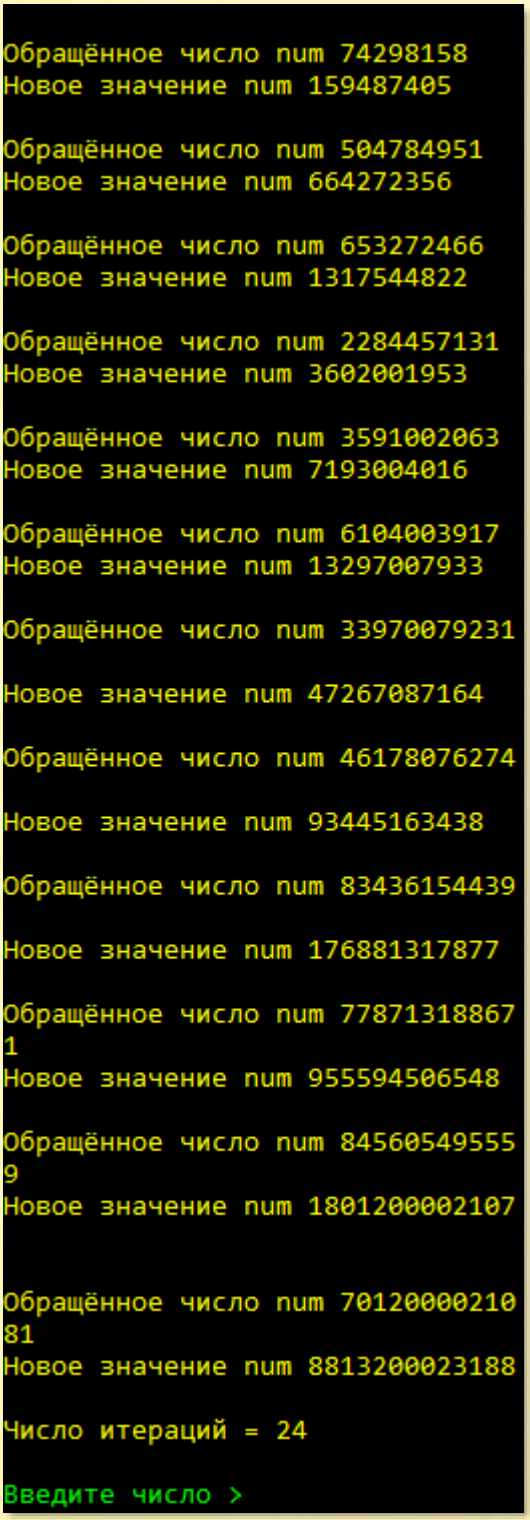
Для переворота чисел и проверки их на палиндромичность в программе используются функции ReverseNum и IsPalindromicNumber:

```
function IsPalindromicNumber(num: BigInteger): bool;
begin
    var rev: BigInteger := 0;
    while (num >= rev) do
    begin
        rev := 10 * rev + num mod 10;
        if (num = rev) then
        begin
            Result := true;
            exit;
        end;
```

```
        num := num div 10;

        if (num = rev) then
        begin
            Result := true;
            exit;
        end;
    end;
    Result := false;
end;


function ReverseNum(num: BigInteger): BigInteger;
begin
    var rev: BigInteger := 0;

    while (num > 0) do
    begin
        rev := rev * 10 + num mod 10;
        num := num div 10;
    end;
    Result := rev;
end;
```

Среди чисел, меньших 10000, самым «вредным» оказалось число 89, которое превращается в палиндромическое только после 24 итераций:





Представленные ниже числа – кандидаты в числа Лишрел:

196, 295, 394, 493, 592, 689, 691, 788, 790, 879, 887, 978, 986, 1495, 1497,1585, 1587,1675,1677, 1765, 1767, 1855, 1857, 1945, 1947, 1997, 2494, 2496, 2584, 2586, 2674, 2676, 2764, 2766, 2854, 2856, 2944, 2946, 2996, 3493, 3495, 3583, 3585, 3673, 3675

Будьте с ними осторожны!

Другие палиндромические диковинки:

Число 10911 становится палиндромическим через 55 шагов: 4668731596684224866951378664

Число 1 186 060 307 891 929 990 после 261 итерации превращается в палиндром, в котором 119 цифр:

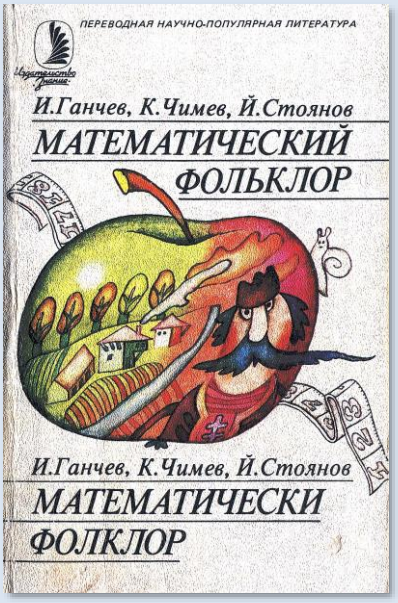
4456266587897643762243784897665387038888478366259842585596343695585248952663874888830783566798487342267346798785662654 4

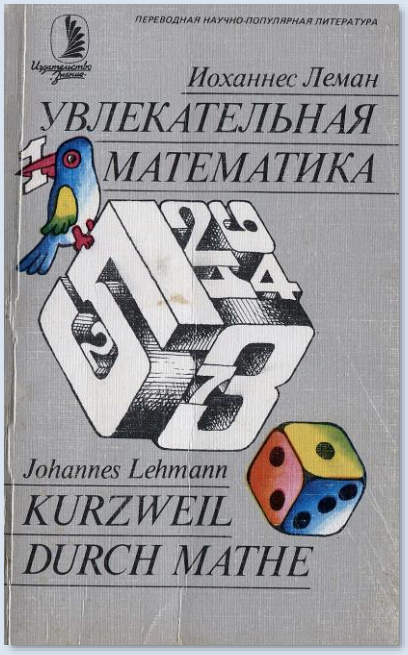
Это число считается мировым рекордсменом. Его нашёл Джейсон Дусетт (Jason Doucette) 30 ноября 2005 года. На его сайте

http://www.jasondoucette.com/worldrecords.html#Most

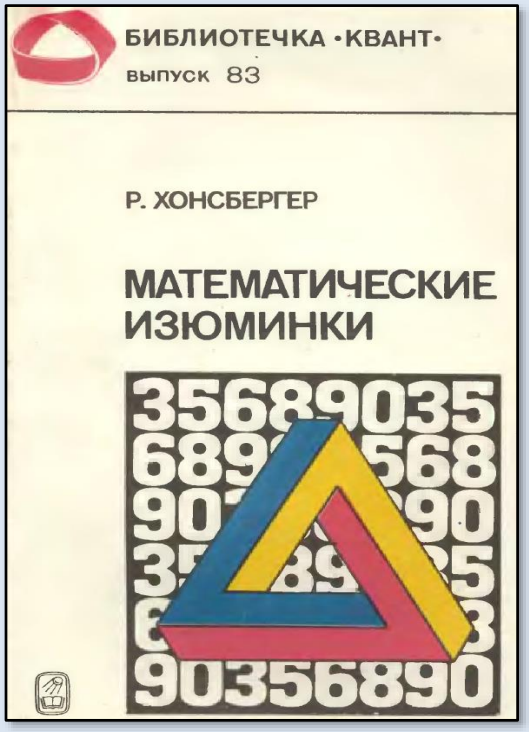
вы найдёте немало интересного о палиндромических числах.

Литература

[Фольклор]	
	Ганчев, Чимев, Стоянов Математический фольклор Знание. Москва, 1985. – 208 с.

[Увлекательная математика]	
	Иоханнес Леман Увлекательная математика Знание. Москва, 1985. – 272 с.

[XP92]



Росс Хонсбергер

Математические изюминки

Наука, 1992. - 176 с.
Библиотечка «Квант». Вып. 83

ISBN 5-02-014406-1

[ГМ72]

Гарднер Мартин

Математические досуги

М.:Мир, 1972. – 495 с.