Интересная задачка. Я, к примеру, не знаю критерия, позволяющего определить количество определённых цифр в заданном диапазоне разрядов иррационального числа. Но мало ли чего я не знаю. Первая сложность - никакие общепринятые форматы чисел ЭВМ не поддерживают точность до 2036 цифр. Ну и ладно. Будем считать золотым сечением   
  
http://www.cyberforum.ru/cgi-bin/latex.cgi?\varphi=\frac%7b\sqrt%7b5%7d+1%7d%7b2%7d=1.6180339887...  
  
или  
  
http://www.cyberforum.ru/cgi-bin/latex.cgi?\varphi=\sqrt%7b\frac%7b5%7d%7b4%7d%7d+\frac%7b1%7d%7b2%7d=\sqrt%7b1.25%7d+0.5  
  
Заметим, что для решения задачи достаточно найти необходимые для решения задачи разряды в числе  
  
http://www.cyberforum.ru/cgi-bin/latex.cgi?\varphi-0.5=sqrt%7b1.25%7d=1.1180339887...  
  
, так как запись этого числа отличается от записи числа золотого сечения только первым разрядом после десятичной точки. Осталось найти, допустим, 2500 знаков вышеуказанного числа, типа, с запасом... Воспользуемся алгоритмом нахождения квадратного корня [http://hijos.ru/2012/04/25/krasivaya...atnogo-kornya/](http://hijos.ru/2012/04/25/krasivaya-modifikaciya-metoda-izvlecheniya-kvadratnogo-kornya/), но... Опять засада... Числа, которые будут фигурировать в программе, могут тоже иметь 2500 разрядов... Однако, применяемое в указанном методе вычитание - простая операция, поэтому тупо можно воспользоваться длинной арифметикой... Для упрощения алгоритма (чтобы сносить только нули) первые два шага выполним вручную. Получим две первые цифры корня: 11, последнее вычитаемое 21, остаток 4. Отсюда и будем плясать. Вот, набросал программку на Паскале на скорую руку... Она насчитала три семёрки в указанном диапазоне. program gold;

const m = 2499;

type nn = array [1..2 \* m] of smallint;

var resid, subtr: nn;

    squar: array [1..m + 1] of smallint;

    i, j, k, dig: integer;

    f: boolean;

//    g: text;  //для печати золотого сечения в файл

procedure addnum(n: byte);

var j: integer;

begin

  subtr[1] := subtr[1] + n; //последнее вычитаемое + n

  for j := 1 to 2 \* m - 1 do

    if subtr[j] >= 10

      then begin

        subtr[j + 1] := subtr[j + 1] + 1;

        subtr[j] := subtr[j] - 10

      end

      else break;

end;

begin

  for i := 1 to 2 \* m do

    begin

      resid[i] := 0;

      subtr[i] := 0

    end;

  for i := 1 to m do squar[i] := 0;

  squar[1] := 1;

  squar[2] := 0; //десятичная точка

  squar[3] := 6; //почему бы и нет? Золотое сечение ищем.

  resid[1] := 4; //остаток

  subtr[2] := 2; //последнее вычитаемое

  subtr[1] := 1;

  for i := 4 to m do //ввычисляем цифирки... номер первого разряда числа = 4

    begin

      for j := 2 \* m downto 3 do resid[j] := resid[j - 2]; //сносим 00

      resid[2] := 0;

      resid[1] := 0;

      addnum(1); //последнее вычитаемое + 1

      for j := 2 \* m downto 2 do subtr[j] := subtr[j - 1]; //последнее вычитаемое \* 10 + 1

      subtr[1] := 1;

      dig := 0; //значение следующуго разряда пока 0

      f := false; //считаем, что разряд неверный

      repeat //вычисляем разряд, разность и остаток

        for j := 2 \* m downto 1 do //сравниваем остаток и вычитаемое

          if resid[j] < subtr[j]

            then begin

              f := false;

              break

            end

            else

              if resid[j] > subtr[j]

                then begin

                  f := true;

                  break

                end;

        if f //если остаток больше вычитаемого,

          then begin //то

            inc(dig); //значение следующего разряда + 1

            for j := 1 to 2 \* m do //вычитаем из остатка вычитаемое

              begin

                resid[j] := resid[j] - subtr[j];

                if resid[j] < 0

                  then for k := j to 2 \* m - 1 do

                    begin

                      resid[k] := resid[k] + 10;

                      resid[k + 1] := resid[k + 1] - 1;

                      if resid[k + 1] >= 0 then break

                    end

              end;

            addnum(2); //последнее вычитаемое + 2

          end

      until not(f);

      subtr[1] := subtr[1] - 2; //восстанавливаем предпоследнеe

      for j := 1 to 2 \* m - 1 do

        if subtr[j] < 0

          then begin

            subtr[j + 1] := subtr[j + 1] - 1;

            subtr[j] := subtr[j] + 10

          end

          else break;

      squar[i] := dig;

    end;

//  assign(g, 'c:\gold.txt'); //для печати золотого сечения в файл

//  rewrite(g);

//  for i := 1 to m + 1 do

//    begin

//      if i = 2

//        then write(g, '.')

//        else write(g, squar[i]:1);

//      if i mod 100 = 0 then writeln(g)

//    end;

//  close(g);

  write('Count of digit "7" in 1991..2036 = ');

  dig := 0;

  for i := 1991 + 2 to 2036 + 2 do if squar[i] = 7 then inc(dig);

  writeln(dig);

  readln

end.

[**Оригинальный метод извлечения квадратного корня**](http://hijos.ru/2012/04/25/krasivaya-modifikaciya-metoda-izvlecheniya-kvadratnogo-kornya/)

25 Апрель 2012, 0:03

[](http://hijos.ru/wp-content/uploads/2012/04/felix.jpg)

Арифмометр Феликс

Об извлечении квадратного корня в столбик я уже писала [здесь](http://hijos.ru/2010/12/22/izvlechenie-kvadratnogo-kornya-v-stolbik/). Сейчас хочу вам предложить модификацию этого метода, которая мне кажется гораздо более простой и красивой. Предложил ее Сергей Валентинович Савич, который и написал мне об этом. Метод был изобретен для арифмометра — механической вычислительной машины, на которой в свое время считали, причем очень активно. Вот что пишет сам Сергей Валентинович: “Схема вычисления квадратных корней была придумана мной в конце 70-х г., и мне хотелось даже опубликовать её описание. Но время арифмометров закончилось, и этот алгоритм остался только в моей памяти. Теперь немного истории. В 1975-76 гг. я учился в Ленинградском топографическом техникуме, и у нас было очень много высокоточных измерений и расчётов. Калькуляторов и ПК тогда ещё не было, всё считали на арифмометрах, а значения функций приходилось брать из толстых 7 — 10-значных таблиц. Тогда у меня и появилась мысль “научить’’ арифмометр вычислять корни. Проштудировал кучу литературы, но ничего подходящего не нашёл. Методы последовательных приближений (дихотомии, Ньютона) отложил как неэкономичные для реализации на арифмометре. Когда я узнал о свойстве арифметической прогрессии нечётных чисел, то решил попробовать на его основе разработать алгоритм извлечения квадратного корня. Наибольшую трудность представляло то, что я никак не мог сообразить, что делать с остатком от вычитания квадрата первой цифры из аргумента. Интуитивно было понятно, что решение где-то рядом. В общем, вертел этот остаток и так, и сяк, и в конечном итоге набрёл на правильное решение.” Как уже говорилось, идея очень красивая. В основе метода лежит следующее свойство суммы nпервых нечетных чисел:

1+3+5+\ldots+(2n-1)=n^2 .

Доказать это легко, если вспомнить формулу суммы nпервых членов арифметической прогрессии. Еще понадобится следующая идея, которая основывается на этом свойстве. Если требуется приписать к какому-то числу aсправа еще одну цифру b, а потом умножить полученное число на эту же цифру, то сделать это можно сложением bпоследовательных нечетных чисел:

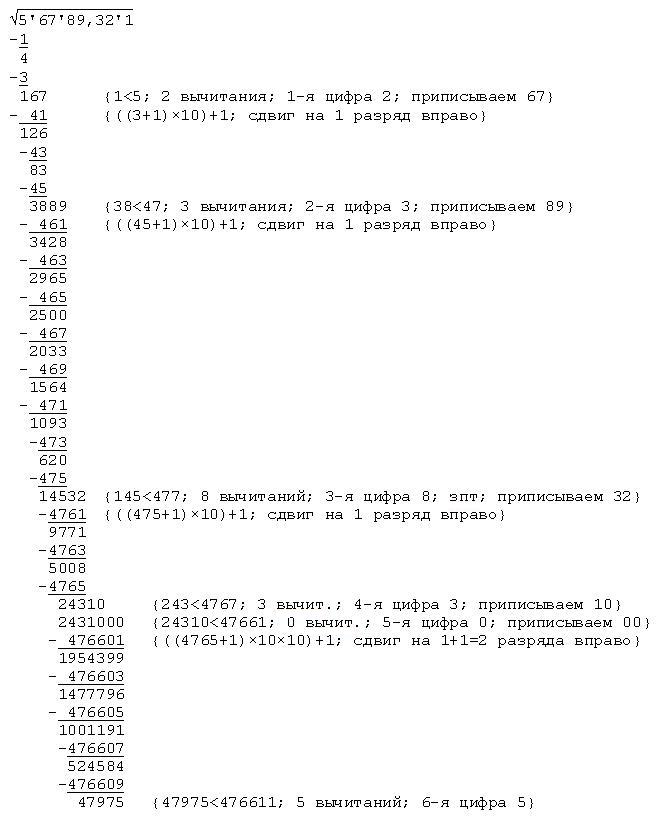
(10a+b)\cdot b=(10a+1)+(10a+3)+(10a+5)+\ldots+(10a+(2b-1)) .

Действительно, давайте раскроем скобки во всех bслагаемых справа и их перегруппируем:

(10a+1)+(10a+3)+(10a+5)+\ldots+(10a+(2b-1))=

=10ab+(1+3+5+\ldots+(2b-1))=10ab+b^2=(10a+b)b .

А теперь можно изменять метод, о котором уже было рассказано. Я опишу каждый шаг, а затем на примере покажу, как он выполняется. **1.** Сначала мы точно так же, как было описано, делим число, из которого будем извлекать квадратный корень, на группы цифр, по две цифры в каждой группе. Целую часть числа (то, что стоит слева от десятичной запятой) разбиваем на группы по две цифры справа налево, а дробную часть (начиная от первой цифры справа от десятичной запятой) — слева направо. Пример давайте рассмотрим тот же, что уже был просчитан. Можно будет сравнить, насколько проще получается алгоритм. Возьмем число 56789,321. Итак, разбиваем цифры этого числа на пары 5^{\prime}67^{\prime}89.32^{\prime}1. **2.** Из первой группы цифр слева вычитаем последовательно нечетные числа 1,3,5,7,\ldots,(2n-1)до тех пор, пока не получится отрицательное число. Запомним последнюю положительную разность, которая получилась — A, и то число, которое при этом вычитали — 2n-1. Если последняя разность равна 0, то корень извлечен точно. В нашем примере можно вычесть два нечетных числа: 5-1=4,4-3=1 .Это означает, что первая цифра корня – 2, остаток A = 1, вычитали при его получении число 2n – 1 = 3. Здесь n— порядковый номер последнего нечётного числа. Далее, к последнему вычитаемому прибавим 1. В нашем случае к 3прибавляем 1, получаем число y = 4. **3.** Теперь к остатку Aсправа припишем следующую группу Cиз двух цифр подкоренного числа, получим число 100A + C. В нашем примере, приписывая следующие две цифры, получим число 167. Из числа 100A + Cснова начинаем вычитать последовательно нечётные числа, предварительно сдвинув вычитаемое на один разряд вправо. Эти нечётные числа формируются следующим образом: первое число, которое будем вычитать, будет число y, к которому справа припишем цифру 1, т.е. 10y + 1, затем 10y + 3; 10y + 5;\ldots, 10y + (2n – 1). Делаем вычитания до тех пор, пока остаток не станет меньше вычитаемого. И снова запомним остаток (это будет A), последнее вычитаемое 10y + (2n – 1)и количество вычитаний n. Если же получили разность 0, и оставшиеся цифры в подкоренном числе — нули, то корень извлечён точно. В нашем примере делаем следующие действия: 167- 41 = 126; 126 – 43 = 83; 83 – 45 = 38; 38 < 47, поэтому вычитания прекращаем. Было выполнено три вычитания, значит, следующая цифра корня — 3. **4.** Если нам нужно посчитать следующие цифры корня, то возьмём снова ту последнюю положительную разность, которую мы запомнили в п. 3, как A, и последнее вычитаемое 10y + (2n – 1), увеличенное на 1– как y. Для разбираемого нами примера получим A = 38, y = 45 + 1 = 46. И повторяем алгоритм, начиная с шага 3. В том случае, если первое вычитаемое 10y + 1больше, чем 100A + C, то следующая цифра корня — 0, так как ни одного вычитания сделать нельзя. В этом случае принимаем 100A + Cза A, 10yза y, и выполняем действия, начиная с шага 3. Если же корень вычислен точно (последняя разность равна 0и оставшиеся цифры справа в подкоренном числе — нули) или корень вычислен с требуемой точностью, то завершаем процесс. Далее я не буду описывать все шаги, просто приведу то, что получается, если все вычисления записать в столбик. Пояснения – в фигурных скобках.

[](http://hijos.ru/wp-content/uploads/2012/04/root2.jpg)

\sqrt{56789,321} = 238,305{точность – 6 значащих цифр, 21 вычитание} Как видно из приведённого описания, этот способ легко может быть формализован и записан в виде программного кода для ЭВМ, причём время выполнения этой программы сопоставимо с временем выполнения операции деления. Сергей Валентинович предложил также некоторые изменения, упрощающие процесс вычислений и позволяющие повысить точность. Так, можно заметить, что после каждого цикла вычитаний цифры последнего вычитаемого, увеличенного на 1 (т.е. y, по нашему соглашению), представляют собой цифры удвоенного значения корня. Поэтому можно не подсчитывать число вычитаний, а просто разделить y, полученное из последнего вычитаемого, на 2. Положение запятой можно определить, руководствуясь следующим правилом: количество цифр целой части результата равно количеству пар цифр в целой части аргумента, каждой паре цифр аргумента соответствует одна цифра результата.

**Примеры:**

аргумент 61 71 , 67 36                                                                                               (1)

результат 7   8 ,    5   6

аргумент   0 , 00 78 93 10                                                                                        (2)

результат  0 ,   0  8   8   8  4   3   1   2

Кроме этого, вдобавок можно получить ещё примерно столько же значащих цифр, сколько уже было получено, простым делением последнего остатка с приписанными неиспользованными парами цифр на последнее вычитаемое, увеличенное на 1. Для описанного нами примера имеем: 47975 / 476610 = 0,10065881…;Здесь шесть цифр 100659являются, с учётом округления последней, верными. Отсюда получаем конечный результат с точностью 12 дес. знаков (!): \sqrt{56789,321} = 238,305100659.Конечно, вычислять вручную с такой точностью значения корней вряд ли целесообразно, но данное правило позволяет для нахождения корней с точностью 5-6 знаков ограничиться вычислением с помощью вычитания нечётных чисел только 3-х цифр.

Ни Сергей Валентинович, ни я нигде не встречали описание данного метода. Если кто-либо из вас где-то об этом читал, напишите, пожалуйста. Заранее большое спасибо.

Когда-то уже довольно давно, когда я училась классе в восьмом, моя учительница [математики](http://hijos.ru/) на кружке показала, как в столбик можно извлекать квадратные корни. Вычислить корень можно с произвольной точностью, найти сколько угодно цифр в его десятичной записи, даже если он получается иррациональным. Алгоритм запомнился, а вопросы остались. Непонятно было, откуда взялся метод и почему он дает верный результат. В книжках этого не было, а может, просто не в тех книжках искала. В итоге, как и многое из того, что на сегодняшний день знаю и умею, вывела сама. Делюсь своим знанием здесь. Кстати сказать, до сих пор не знаю, где приведено обоснование алгоритма)))

Итак, сначала на примере рассказываю, “как работает система”, а потом объясняю, почему она на самом деле работает.

Возьмем число 56789,321(число взято “с потолка”, только что в голову пришло).

**1.** Разбиваем его цифры на пары: те, что стоят слева от десятичной запятой, группируем по две справа налево, а те, что правее – по две слева направо. Получаем 5` 67` 89,32` 1.

**2.** Извлекаем квадратный корень из первой группы цифр слева – в нашем случае это 5(ясно, что точно корень может не извлекаться, берем число, квадрат которого максимально близок к нашему числу, образованному первой группой цифр, но не превосходит его). В нашем случае это будет число 2. Записываем 2в ответ – это старшая цифра корня.

**3.** Возводим число, которое стоит уже в ответе — это 2— в квадрат и вычитаем из первой слева группы цифр – из числа 5. В нашем случае остается 1.

**4.** Приписываем справа следующую группу из двух цифр: 167. Число 2, которое уже стоит в ответе, умножаем на 2, получаем 4.

**5.** Теперь следите внимательно. Нам нужно к числу 4справа приписать одну цифру b, и число \overline{4b}умножить на b, то есть на ту же самую приписанную цифру. Результат должен быть как можно ближе к 167, но опять-таки не больше этого числа. В нашем случае это будет цифра 3, ее записываем в ответ рядом с 2, справа. Это следующая цифра в десятичной записи нашего квадратного корня.

**6.** Из 167вычитаем произведение 43\cdot3=129, получаем 38.

**7.** Далее повторяем знакомые операции: приписываем к 38справа следующую группу цифр 89, 23умножаем на 2, к полученному числу 46приписываем справа одну цифру, такую, чтобы при умножении на нее получилось число, меньшее 3889, но наиболее близкое к нему – это цифра 8– следующая цифра в десятичной записи корня.

**8.** Далее у нас в числе стоит десятичная точка, ставим такую же в результате после цифры 8. Продолжаем процесс, снося по две цифры после точки. Ясно, что можно сносить и два нуля.

Вычисления запишутся следующим образом:

\begin{tabular}{c@{}r@{}r@{}c@{}c@{}c}<br />
\multicolumn{5}{l}{$\sqrt{5 ` 67 ` 89.32 ` 1}$}&$=238.30\ldots$\\<br />
\ \ 4 & & & & &\\<br />
\cline{1-3}<br />
\ \ 1 & 67 & & & &\\<br />
\ \ 1 & 29 & & & &\\<br />
\cline{1-4}<br />
\ \ & 38 & 89 & & &\\<br />
\ \ & 37 & 44 & & &\\<br />
\cline{2-5}<br />
\ \ & 1 & 45 & 32 & &\\<br />
\ \ & 1 & 42 & 89 & &\\<br />
\cline{2-5}<br />
\ \ & & 2 & 43 & 1000 &\\<br />
\ \ & & 2 & 38 & 3025 &\\<br />
\cline{3-5}<br />
\ \ & & & 4 & 7975 &\\<br />
\ \ & & & & $\ldots$ &<br />
\end{tabular}<br />
\begin{array}{l}<br />
43\cdot3=129,\\<br />
468\cdot8=3744,\\<br />
4763\cdot3=14289,\\<br />
476605\cdot5=2383025.<br />
\end{array}

А теперь обещанное объяснение. Алгоритм основан на формуле

(10a+b)^2=100a^2+20ab+b^2=100a^2+(20a+b)b.

Первый раз вычитаем квадрат, дальше, приписывая по одной цифре к результату, к числу под корнем, тем самым, приписываем две десятичных цифры. Отсюда разбиение на пары (видно из формулы). Вычтя квадрат, необходимо вычитать дальше числа вида (20a+b)b, где 2a– удвоенный известный на данный момент результат, приписывая к нему цифру, получаем 20a+b, умножаем на эту же самую цифру, имеем (20a+b)b. Вот и все!